TRUST-BASED COOPERATIVE GAMES AND CONTROL STRATEGIES FOR AUTONOMOUS MILITARY CONVOYS

DISSERTATION FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY IN
ELECTRICAL AND COMPUTER ENGINEERING

DARIUSZ G. MIKULSKI

OAKLAND UNIVERSITY

2013

Report Documentation Page

Form Approved
OMB No. 0704-0188

1. REPORT DATE **01 OCT 2013**	2. REPORT TYPE **Dissertation**	3. DATES COVERED **10-03-2012 to 16-09-2013**

4. TITLE AND SUBTITLE
TRUST-BASED COOPERATIVE GAMES AND CONTROL STRATEGIES FOR AUTONOMOUS MILITARY CONVOYS

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)
Dariusz Mikulski

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
U.S. Army TARDEC,6501 East Eleven Mile Rd,Warren,Mi,48397-5000

8. PERFORMING ORGANIZATION REPORT NUMBER
#24307

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
U.S. Army TARDEC, 6501 East Eleven Mile Rd, Warren, Mi, 48397-5000

10. SPONSOR/MONITOR'S ACRONYM(S)
TARDEC

11. SPONSOR/MONITOR'S REPORT NUMBER(S)
#24307

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited

13. SUPPLEMENTARY NOTES

14. ABSTRACT

According to the Brookings Institute, the leading cause of death for U.S troops in the War in Afghanistan was due to improvised explosive devices (IEDs) [82]. This finding has also been the general case in Southwest Asia in recent years, where enemy combatants have used IEDs as a quick and deadly adaptation to US strategies and tactics [127]. Early in these conflicts, IEDs were jury-rigged homemade bombs that while deadly, could be avoided with increased awareness. But enemy combatants 2 quickly adapted by developing more sophisticated explosives, often with timing devices, pressure switches, and even wireless triggers. In addition, enemy combatants became more difficult to detect due to their knowledge of the local terrain and their ability to mix with civilian populations. In response to these more advanced IED attacks, U.S. troops adjusted their own tactics and strategies, which ultimately required them to put more trust into local allies and new equipment (such as up-armored vehicles, electronic jammers, and robots). And while this did not necessarily imply that any warfighter was safer than before, the trust helped warfighters deal internally with wartime uncertainties so that they could continue their duties and focus on mission objectives in this hostile environment.

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT **unclassified**	b. ABSTRACT **unclassified**	c. THIS PAGE **unclassified**	**Public Release**	**301**	

Standard Form 298 (Rev. 8-98)
Prescribed by ANSI Std Z39-18

TRUST-BASED COOPERATIVE GAMES AND CONTROL STRATEGIES FOR AUTONOMOUS MILITARY CONVOYS

by

DARIUSZ G. MIKULSKI

A dissertation submitted in partial fulfillment of the
requirements for the degree of

DOCTOR OF PHILOSOPHY IN
ELECTRICAL AND COMPUTER ENGINEERING

2013

Oakland University
Rochester, Michigan

Doctoral Advisory Committee:

Edward Y. L. Gu, Ph.D. (Co-Chair, Oakland University), Professor
Frank L. Lewis, Ph.D. (Co-Chair, University of Texas in Arlington), Professor
Darrell P. Schmidt, Ph.D., Professor
Guangzhi Qu, Ph.D., Associate Professor
Osamah A. Rawashdeh, Ph.D., P.E., Associate Professor

To my children, whom I dearly love.

“As soon as you trust yourself, you will know how to live.” – J.W. von Goethe

ACKNOWLEDGMENTS

No one has ever achieved greatness alone, nor succeeded without experiencing failure. That is why I am humbled by and grateful to the following people who have provided me with encouragement, inspiration, and support during my dissertation research. Firstly, I am grateful to my precious wife, Kimberly, who provided consistent support, sacrifice, and encouragement at home. My appreciation for this cannot be understated. I am also grateful to my co-chairs, Dr. Edward Gu and Dr. Frank Lewis, for their mentoring and guidance throughout a difficult, but rewarding, dissertation process. Both challenged me in different ways to do my best, but together provided me with helpful advice, collaborative opportunities, and creative freedom. I am also grateful to my committee members, Dr. Darrell Schmidt, Dr. Guangzhi Qu, and Dr. Osamah Rawashdeh, for challenging me to apply my theories to relevant engineering problems, which influenced my thought processes, and subsequently, the overall direction of the research. I am also grateful to: Dr. Greg Hudas, Dr. Jim Overholt, and Dr. David Gorsich for their encouragement to pursue ILIR funding from ASA(ALT), which supported this research; Dr. Paul Rogers, Mr. Thom Mathes, and Mr. Paul Skalny for their blessings to participate in the Army's Long-term Training Program in 2010; my brother, Christopher, who spent countless hours proofreading this dissertation; and my parents, Aleksandra and Paweł, whose unyielding interest in my research progress provided me with the persistent motivation to continue moving forward.

Dariusz G. Mikulski

PREFACE

My journey towards this dissertation started seven years ago, shortly after finishing my Masters degree at Oakland University (OU) in 2006. Newly engaged to my wife, Kimberly, I found myself in the throes of wedding planning – an experience worth doing only once – and pondering deeply about my future. My contemplations led me to accept, among other things, a reality that I likely knew for some time – that I love doing scientific research. I had a wonderful experience writing my Master's thesis, and since finishing it, felt a chronic yearning to rekindle the research process that led to its development. Needless to say, I applied to the Ph.D. program at the School of Engineering and Computer Science (SECS), and was set to start my course work in the New Year.

Around this time, however, my employer, the U.S. Army Tank-Automotive Research Development and Engineering Center (TARDEC), who supported my Master's thesis research, was implementing a reorganization strategy across all of its business groups. And unfortunately for me, one of its causalities was the dissolution of my research group, the Automotive Research Team. But while disappointing at the time, it turned out to be an opportune twist of fate. Within days, I was offered a unique opportunity by Mr. Michael Letherwood to be a founding member of the University Partnerships Team within the National Automotive Center at TARDEC. And while this was not a research group like my previous team, it gave me the opportunity to travel frequently and meet with professors and researchers at many national universities.

In November of 2007, I visited the University of Texas in Arlington (UTA), where I met Dr. Frank Lewis for the first time. Our meeting was actually unplanned, as I was at UTA for other business. But we spoke long enough for me to realize that Dr. Lewis's research projects were unique and advanced, and certainly of great potential value to TARDEC's strategic robotics thrust. As such, I recommended Dr. Lewis's work to Dr. Greg Hudas and Dr. Jim Overholt at the TARDEC Ground Vehicle Robotics (GVR) group. This eventually resulted in a new joint research project between TARDEC GVR and UTA.

Over the next year, I monitored this project and along the way developed a strong professional relationship with Dr. Lewis. I also became increasingly fascinated with robotics research, particularly in the area of multi-robot cooperative teaming. At the time, I knew TARDEC did not have any researchers working in this area [36]. And it was this gap that ultimately motivated me to select it as the general research domain for my Ph.D. dissertation.

To chair my doctoral advisory committee, I sought out Dr. Edward Gu at OU, who I knew to be an experienced engineer and well-respected by his peers. And after our first meeting, I came to also know of his graciousness and humility. During this inaugural meeting, I suggested the idea to invite Dr. Lewis, a professor from a different university, to our committee as a "co-chair." I gathered it was an unusual request, given Dr. Gu's inquisitive reaction. But after a short discussion on the matter, Dr. Gu gave me his complete support. I later learned how unprecedented this support was, since according to the OU Office of Graduate Study, this type of request had never happened before in the history of Oakland University. And indeed, they cited this fact as one of the reasons for initially rejecting my doctoral committee application. However, thanks to a wonderfully written letter of support from Dr. Bhushan Bhatt, who was serving as Dean of SECS, the Office of Graduate Study reconsidered my committee application, and ultimately approved it.

It was around this same time that I sought out and was granted permission by my employer to participate in the U.S. Army Competitive Professional Long-Term Training program. This program allowed me to both work exclusively at OU on independent robotics research for one year and satisfy OU’s one-year residency requirement for doctoral students (which I chose to serve during the 2010 calendar year). During this year, I poured over literature about control architectures, game theory, graph theory, agent-based learning, and mobile robotics – much of which was foreign to me. But over time, I began to understand it well enough to realize all of the potential vulnerabilities military robots could be exposed to in teaming applications. These vulnerabilities, however, were not conceptually dissimilar from those vulnerabilities faced by people or animals working together in teams. And it was this parallel that led me to consider the heuristic that biological agents use to deal with vulnerabilities and justify cooperative relationships – trust.

The literature on trust, from both a philosophical and computational perspective, is extremely rich and supported by decades of work from many brilliant thinkers. Despite this, however, I was somewhat surprised by the number of computational trust researchers who narrowly defined interactions between agents on communication alone. In other words, I found that interactions with other agents were largely based on asking about or listening to what others were saying about themselves or others. And because of this, trust models were often tailored to information-based or transactional systems, like wireless sensor networks, social networks, and internet applications. I was also somewhat intrigued by the manner in which trust research was being conducted within the broader robotics research domain. Much of the trust research was associated with human-robot interaction research, with the goal to understand how humans develop trust towards robots. Indeed, I was unable to find any work related to trust cultivation in the reverse direction; that is, how robots may develop trust towards humans (or other robots).

I personally believe that a robot can be more than an information system, even when networked together with other robots. A robot, after all, is also a mechanical vessel with moving parts of its own that is able to influence, or be influenced by, the physical environment. And if one designs a robot to behave autonomously in a dynamic, unstructured, and open physical environment, then its behaviors may begin to resemble those of a living creature. Thus, it is not far-fetched to consider humans forming pseudo-relationships with robots, much in the same way as they would with a pet [133]. In a human-pet relationship, mutual trust can be thought of as the glue that binds the human and pet for cooperative coexistence.

Serendipitously, I became very aware of the interplay between trust and cooperative coexistence while training my new dog, Abby, in 2010. And it was this experience that helped me to narrow in on my specific dissertation topic. At the start of Abby's training, I established a very simple contract with her; that all validated understandings of my commands would be rewarded with a treat. And over time, after many successful executions of our contract, a mutual trust emerged that allowed me to replace the physical treat with verbal praise. This is not unlike friendship, as close friends tend to do favors for each other without the expectation of an equal exchange or tangible reward. And what I believe, indeed, occurred during the training process was the formation of a friendship between Abby and me. I learned that I cannot make Abby do something that she does not want to do. But with enough trust, I can expect her to do something that she naturally would not do. And it was this realization that inspired me to consider ideas surrounding trust-based control for robotic systems. I was forced to confront and reconsider assumptions that I, and many others, often take for granted in human-robot interactions. Could a robot rightfully refuse commands or withhold capabilities from an operator? Could a robot influence the manner in which the operator controls it? Could the operator behave as if it is in a relationship with the robot rather than the robot behave as if it is an extension of the operator? Could it be possible for a robot to trust its operator in a similar way that a pet trusts its owner?

My search for answers to these, and many other, questions began within the mathematics of cooperative game theory, which seeks to understand how self-interested agents can combine to form effective teams. Using its mathematical formulations as a foundation, I developed the cooperative trust game, which can predict coalition formation from trust-based interactions. Within my framework, I introduced the new concepts of trust synergy and trust liability, which juxtapose the potential gains and losses in a trusting relationship. These constructs serve as the cornerstones of the framework, and formally embody the philosophical notions of instrumental value and vulnerability in trusting relationships. But while the cooperative trust game may coarsely predict the outcomes of trust-based interactions in coalitions, it cannot model the dynamics of a trust-based interaction at the agent level. For that, I looked to the computational trust research domain, which contained many trust models that could directly “plug-in” to cooperative trust game framework to demonstrate coalition formation dynamics over time. However, I was disappointed to discover that the literature had little to say about military robotics applications with those trust models.

Alas, my long-term training at OU ended in January of 2011, and upon my return to TARDEC, I was assigned to the Robotic Technologies Team within GVR. At that time, my new team leader, Dr. Robert Kania, briefed me in detail about, among other things, the TARDEC Convoy Active Safety Technology (CAST) program, which won the Army Research and Development Achievement Award in 2009 [119]. The aim of CAST was to improve convoy operations with the installation of a small kit in the cab of a tactical vehicle. The kit connects actuators to the steering wheel, gas pedal, and brake pedal, and uses various sensors (such as radar, lidar, and electro-optical/infrared cameras) to sense the local environment. I was instantly intrigued by this program, not only because of its well-defined multi-agent military application, but also because of the technology’s potential to be fielded in the near term. After that briefing, I decided to make the autonomous military convoy my application focus for this dissertation. I took it upon myself to apply the convoy concept to the cooperative trust game, thereby creating the convoy trust game and successfully proving the highest payoff solution for the *N*-agent convoy trust game.

Following the work with the convoy trust game, I started to feel my research interests slowly shifting away from game theory research and moving closer towards computational trust research. I credit this shift to the rediscovery of a paper that Dr.

Lewis gave me more than a year earlier. The paper was about the work he and his student, Prasanna Ballal, did with trust-based collaborative control in teams [14]. They described a bilinear local voting consensus protocol for trust establishment and used it to invoke flocking behaviors and formations in a distributed network of agents. I felt optimistic that I could adapt their work to control an autonomous convoy, and as such, immersed myself in the study of the consensus problem to better understand what they did. However, in time, I discovered a problem that I could not avoid if I intended to use their methods in a practical implementation – their ad-hoc random initialization of trust values for the bilinear local voting consensus protocol.

Naturally, I searched the computational trust literature for a suitable solution. But as I recall, I felt frustrated by the clash between the available trust models and my vision of how trust "should" work in robotic contexts. I desired a trust model that was simple, meaningful, and applicable to robots. And this desire ultimately led me to develop my own trust model, which I named RoboTrust. In short, RoboTrust sets the trust value equal to the smallest value in a set of maximum-likelihood estimates that are based on different historical observations. The selection of which historical observations are considered is determined by the trust model's two parameters: tolerance (τ) and confirmation (c). Observations within the history are derived from results of a user-defined acceptance function, which maps a multi-dimensional feature space (i.e. sensor data) to a binary codomain. With RoboTrust, I was able to eliminate the need for the bilinear local voting consensus protocol to update the trust dynamics in a distributed agent network. I demonstrated this fact with a series of case studies using

RoboTrust within a distributed, discrete-time, trust-based consensus protocol that converged to agreement for a decision vector.

Following these case studies, which I primarily used to characterize the behavior of RoboTrust, I returned to my original intention of developing trust-based control for an autonomous convoy. However, the trust-based controller that is presented in this dissertation is drastically different from my original plan to adapt the Ballal-Lewis collaborative control scheme. This is in large part due to my participation on the TARDEC source selection board for the Autonomous Mobility Appliqué System (AMAS) Joint Capability Technology Demonstration (JCTD) program – the follow-on program to CAST. The AMAS JCTD request for proposal called for a scalable autonomy solution that is platform-agnostic and is able to perform multiple tasks, such as convoy, security, and reconnaissance. As such, I had a rare opportunity to review various proposals from different companies, who each had different solutions to the autonomous convoy problem. While reading these proposals, I came to an important realization – that it would be an impractical design decision to tightly couple RoboTrust within an arbitrary convoy control solution in the manner that I coupled it within the trust-based consensus protocol. Rather, a better and more flexible control architecture would use RoboTrust within a high-level decision maker to automatically switch between different low-level control modes based on different levels of trust for different contexts. That way, the low-level control solutions can be developed and maintained independently of the system that evaluates the high-level context of the current observations. To prove the feasibility of this architecture, I developed a convoy simulation application in Matlab and demonstrated how my trust-based controller correctly responds to badly behaving vehicles during simulated convoy missions. Using these results as a springboard, future work intends to incorporate RoboTrust onto a physical robotics platform for intelligent mobility in cooperative heterogeneous teams.

Alas, my dissertation is finished. And looking back over the last seven years, I am amazed at how unexpected inspiration and chance encounters have shaped its development. I am convinced that my studies into trust have had a profound influence on my personal worldview. In particular, I have come to believe that vulnerabilities, in of themselves, are not things that should necessarily be isolated or eliminated from life. Rather, it is people or entities who willingly exploit those vulnerabilities that should be isolated or eliminated. After all, being vulnerable to betrayal is the only way one can begin to cultivate trust in a relationship. Without this vulnerability, trust can never grow, and the potential synergistic gains from a relationship can never be realized. Yet, even if one is willing to be vulnerable, trust may still not grow in a relationship if the other is unwilling to also be vulnerable. Hence, mutual vulnerability governed by a mutual agreement of acceptable behavior establishes the foundations of a trusting relationship. Ultimately, though, it is the character of the individuals within the trusting relationship that determines how mutual trust evolves.

ABSTRACT

TRUST-BASED COOPERATIVE GAMES AND CONTROL STRATEGIES FOR AUTONOMOUS MILITARY CONVOYS

by

Dariusz G. Mikulski

Co-Chairs: Edward Y.L. Gu, Ph.D. and Frank L. Lewis, Ph.D.

The future presence of autonomous military robots in heterogeneous teams will introduce new trust-based vulnerabilities that previously did not exist in homogeneous human teams. Among these vulnerabilities is their exposure to cyber attacks, which can disrupt and/or take over these systems. Evidence exists that other nation-states are actively developing their cyber attack capabilities to break into U.S. military unmanned systems.

Given this problem, our general research goal was to determine the feasibility of computational trust as a defensive capability against unacceptable behaviors in autonomous multi-agent systems. To meet this goal, we sought to develop new or improved computational trust models, algorithms, and frameworks for trust cultivation, aggregation, and propagation in distributed teams. Since autonomous convoy operations are expected to be one of the near-term, large-scale applications of autonomous military technologies, we chose it to be the application focus of our research.

Our work produced two major results. The first was the cooperative trust game – a new mathematical framework to predict coalition formation in response to trust-based interactions. Using this theory, we developed the convoy trust game and proved that the most optimal trust payoff in centralized convoys occurs when the lead vehicle acts as the trusted third-party for all follower vehicles. For decentralized convoys, we discovered that the trust payoff can be maximized if agents view immediate leaders and followers as surrogates for the whole system of agents in front and behind them, respectively.

The second major result was the development of the RoboTrust model – a new computational trust model that assigns a trust value equal to the smallest value in a set of maximum-likelihood estimates based on different historical observations. RoboTrust explicitly separates the context of an observation and the actual trust calculation, providing significant advantages for engineering management and platform deployment over other trust models with tightly coupled contexts and calculations. We applied RoboTrust within two different problems, namely the consensus problem and the autonomous convoy soft security problem. Evidence exhibited in this dissertation allows us to conclude that trust-based control using RoboTrust provides for a feasible soft security solution against unacceptable vehicle behaviors in autonomous military convoys.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF FIGURES – Continued

LIST OF FIGURES – Continued

NOMENCLATURE

Acronyms

ADAPT	Ames Research Center Advanced Diagnostics and Prognostics Testbed
AFOSR	Air Force Office of Scientific Research
AGM	Alchourrón-Gärdenfors-Makinson (postulates)
AMAS	Autonomous Mobility Appliqué System
ARL	Army Research Laboratory
ASA(ALT)	Assistant Secretary of the Army for Acquisition, Logistics, and Technology
CAST	Convoy Active Safety Technology
CBA	Conflicting Behavior Attack
CCG	Constrained Coalitional Game
CCRI	Cross-cutting Research Issue
CF	Closest Follower
CL	Closest Leader
EOARD	European Office of Aerospace Research and Development
EPS	Electrical Power System
FIPA	Foundation for Intelligent Physical Agents
FRA	False Recommendation Attack
GVR	Ground Vehicle Robotics
GUI	Graphical User Interface

NOMENCLATURE – Continued

HFOV	Horizontal Field of View
ID	Identification
IED	Improvised Explosive Device
ILIR	In-House Laboratory Independent Research
JCTD	Joint Capability Technology Demonstration
KB	Knowledge-Belief
MANET	Mobile Ad Hoc Network
NAC	National Automotive Center
NASA	National Aeronautics and Space Administration
NS-CTA	Network Science Collaborative Technology Alliance
OOA	On-Off Attack
OU	Oakland University
PSO	Particle Swarm Optimization
ROS	Robotic Operating System
SECS	School of Engineering and Computer Science
TARDEC	Tank Automotive Research Development and Engineering Center
TPA	Technology Program Agreement
TRAVOS	Trust and Reputation model for Agent-based Virtual OrganizationS
UTA	University of Texas in Arlington
VCG	Vickrey-Clarke Groves (mechanism)
VGA	Video Graphics Array

NOMENCLATURE – Continued

Mathematical Symbols

$\cdot$	Placeholder
$\binom{\beta}{\alpha}$	Binomial coefficient, where $\alpha, \beta \in \mathbb{N}$ and $\alpha \leq \beta$
$\lvert\cdot\rvert$	Cardinality of a set or absolute value of a scalar
$[\cdot,\cdot]$	Closed Interval
$\setminus$	Complement
$\wedge$	Conjunction operator
$\cdots, \vdots, \ddots$	Ellipsis (horizontal, vertical, and diagonal) to mean “and so forth”
$\hat{\cdot}$	Estimated value
$\{\cdot\}$	Extension definition of a set
$\lfloor\cdot\rfloor$	Floor function
$\circ$	Hadamard product
$\langle\cdot,\cdot\rangle$	Inner product
$\cdot^*$	Notation for an intermediate scalar, vector, or matrix in some calculation
∞	Infinity
$\cap$	Intersection
$\Longrightarrow$	Logical entailment
$\lVert\cdot\rVert$	Magnitude of a vector
$\rightarrow$	Material conditional

NOMENCLATURE – Continued

$[\cdot]_{m\times n}$	Matrix with m rows and n columns
$\square$	Modal operator for “necessarily”
$\lozenge$	Modal operator for “possibly”
$\neg$	Negation operator
$\subset$	Proper subset
$[\{\cdot\}_i]_{i=1}^{x}$	Sequence consisting of x indexed elements
$\in$	Set membership
$\subseteq$	Subset
$[\cdot]^{\mathrm{T}}$	Transpose of matrix
$\cup$	Union
$\forall$	Universal quantification
$\{\cdot\}^n$	Vector space of n-tuples over the field given by the set
$\emptyset$	Empty set
$\boldsymbol{A}$	Adjacency matrix
$\underset{x}{\arg\max}$	Argument x of the maximum argument
$a_R(S)$	Altruistic contribution between agents in R and $S\backslash R$, such that $R \subseteq S \subseteq N$
B	Modal operator for belief
$B_i\varphi$	Modal logic notation for “agent i believes that φ”
c	Confirmation parameter for RoboTrust
c_{max}	Max confirmation; a user-defined limit for the confirmation parameter

$c_R(S)$	Competitive contribution of agents in R to S, such that $R \subseteq S \subseteq N$
$\boldsymbol{D}$	Degree matrix
d_i	Minimum following distance of vehicle i (in meters)
$\frac{d}{dx}$	Leibniz's notation for the first derivative with respect to x
Δ_i	Weighted degree of all outgoing edges of i
δ_{ij}	Measure of agent j's willingness to cooperate with agent i during a consensus process
E	Finite set of all directed edges in a graph
ϵ	Step-size on the open interval $(0,1)$ for consensus protocol
F	Finite set of feature space attributes (or dimensions)
$f(\boldsymbol{q}, q)$	Scaling function that outputs the multiple to ensure $\|\boldsymbol{q}\| \leq q$
G	Modal operator for goal
$\mathcal{G}$	Graph
$\mathit{\Gamma}$	Game
$\boldsymbol{h}_i$	Heading of vehicle i
$\boldsymbol{I}$	Identity matrix
$\mathcal{I}_i$	Set of possible worlds from the point of view of agent i in the n-agent partition model
$J_i^{(m)}$	Finite set of the mth-neighbors of agent i
$K_i\varphi$	Modal logic notation for "agent i knows that φ"
k	Time step

NOMENCLATURE – Continued

$l(A)$	Trust liability set function, where $A \subseteq N$
$\mathbb{L}$	Language for the n-agent partition model
$\boldsymbol{\Lambda}$	Trust liability matrix
$\boldsymbol{\mathcal{L}}(\theta\|\boldsymbol{x})$	Likelihood of parameter θ given outcomes $\boldsymbol{x}$
$\lim_{k\to\infty}(\cdot)$	Limit as k approaches infinity
$\log(\cdot)$	Common logarithm
$\boldsymbol{L}$	Graph Laplacian
λ	Eigenvalue
$\hat{\lambda}$	Perron root
$\mathcal{M}$	N-agent partition model
$\max(\cdot)$	Maximum argument
$\min(\cdot)$	Minimum argument
mod	Modulo operation
m_i	Mass of vehicle i (in kilograms)
$m_R(S)$	Marginal contribution of agents in R to S, such that $R \subseteq S \subseteq N$
N	Finite set of all agents
N_X	Finite set of first-neighbors of the agents in set $X \subseteq N$
$\mathbb{N}$	Natural numbers $\{0, 1, 2, \cdots\}$
$\mathbb{N}^+$	Positive natural numbers $\{1, 2, \cdots\}$
$\mathcal{N}(\mu, \sigma^2)$	Normal distribution with bias μ and variance σ^2

$O(\cdot)$	Big-O notation
$\boldsymbol{P}$	Perron matrix
$P_x(\cdot)$	Probability distribution that depends on a parameter x
$\boldsymbol{p}_i$	Linear momentum of vehicle i (in Newtons)
p_i	Maximum impulse force of vehicle i (in Newtons per second)
Π	Interpretation function to determine which sentences in the language are true in which worlds within the n-agent partition model
π	Pi
$\mathbb{R}$	Set of real numbers
r_i	Sensor range of vehicle i (in meters)
ρ	Information function
s_i	Maximum speed of vehicle i (in meters per second)
$s(A)$	Trust synergy set function, where $A \subseteq N$
$\boldsymbol{\Sigma}$	Trust synergy matrix
$\boldsymbol{T}$	Trust matrix
$\boldsymbol{T}_{ij}^{(\mathcal{F})}$	Follower trust value of vehicle i towards its follower j
$t_i^{(\mathcal{F})}$	Follower trust threshold of vehicle i
$\boldsymbol{T}_{i\ell}^{(\mathcal{L})}$	Leader trust value of vehicle i towards its closest leader ℓ
$t_i^{(\mathcal{L})}$	Leader trust threshold of vehicle i
τ	Tolerance parameter for RoboTrust

θ_i	Sensor horizontal field of view of vehicle i (in degrees)
$\tan^{-1}$	Inverse tangent
$\mathcal{U}(a, b)$	Uniform distribution on the interval $[a, b]$
$\boldsymbol{u}_i$	Control signal vector of agent i
$u_R(S)$	Real-value payoff set function perceived by the agent in R when the Agents in S cooperate, where $R \subseteq S \subseteq N$
v_i	Vehicle state parameter of vehicle i
$v(A)$	Real-value payoff set function, where $A \subseteq N$
$v_M(A)$	M-marginal game, where $A \subseteq N$
W	Non-empty, finite set of waypoints
$\boldsymbol{W}$	Weighted adjacency matrix
$\boldsymbol{w}_i$	Current waypoint of vehicle i
w_i	Waypoint radius of vehicle i
ω_i	Current index on the waypoint path for vehicle i
$\mathcal{W}$	Worlds used to specify the state of affairs of a situation or environment in the n-agent partition model
x	**Scalar** (lowercase letter)
$x^{(i)}$	Notation for a scalar identified by i
x_i	Notation for a scalar belonging to i
x_{ij}	Notation for a scalar belonging to i regarding j
$\boldsymbol{x}$	**Vector** (lowercase letter, bold-face)

$\dot{\boldsymbol{x}}$	Newton’s notation to represent the first time derivative of vector $\boldsymbol{x}$
$\boldsymbol{x}(k)$	Notation for the state vector $\boldsymbol{x}$ at time step k
$\boldsymbol{x}^{(i)}$	Notation for a vector identified by i
$\boldsymbol{x}_i$	Notation for a vector belonging to i
$\boldsymbol{x}_{ij}$	Notation for a vector belonging to i regarding j
$\boldsymbol{x}_{(i)}$	Notation for the value indexed by i in the vector $\boldsymbol{x}$
$\boldsymbol{X}$	**Matrix** (uppercase letter, bold-face)
$\boldsymbol{X}^m$	Notation for mth power of a square matrix $\boldsymbol{X}$
$\boldsymbol{X}^{-1}$	Notation for inverse of a square matrix
$\boldsymbol{X}^{(i)}$	Notation for a matrix identified by i
$\boldsymbol{X}_{ij}$	Notation for the value at row i and column j in matrix $\boldsymbol{X}$
X	**Set** (uppercase letter)
$X^{(i)}$	Notation for a set identified by i
X_i	Notation for a set belonging to i
$X_{(i)}$	Abusive notation for the value in set X at index i
xy	2-dimensional Euclidean plane
$Z(k, F)$	Set of observations at time k for the features in F
$z(k)$	Acceptance function
$\boldsymbol{z}_{ij}^{(c)}$	Sequence of the $c + 1$ most recent observations about j from an acceptance function from i
$\mathbb{Z}$	Set of positive and negative integers

CHAPTER ONE

INTRODUCTION

Synopsis

This chapter introduces the research in this dissertation by framing it within the context of an autonomous convoy soft security problem. We describe this problem specifically in Section 1.1, justifying the existence of new trust-based vulnerabilities that previously did not exist in homogeneous manned convoy operations. In Section 1.2, the specific problem is placed within the context of larger research domains, namely computational trust and network science, in order to show how it fits into broader research initiatives. Finally, we state our research objective and contributions in Sections 1.3 and 1.4, respectively. We also provide a publications list in Section 1.5 to indicate our original, peer-reviewed research results during the dissertation research period.

1.1 Specific Problem: Trust Vulnerabilities in Autonomous Military Convoys

According to the Brookings Institute, the leading cause of death for U.S troops in the War in Afghanistan was due to improvised explosive devices (IEDs) [82]. This finding has also been the general case in Southwest Asia in recent years, where enemy combatants have used IEDs as a quick and deadly adaptation to US strategies and tactics [127]. Early in these conflicts, IEDs were jury-rigged homemade bombs that, while deadly, could be avoided with increased awareness. But enemy combatants quickly adapted by developing more sophisticated explosives, often with timing devices, pressure switches, and even wireless triggers. In addition, enemy combatants became more difficult to detect due to their knowledge of the local terrain and their ability to mix with civilian populations. In response to these more advanced IED attacks, U.S. troops adjusted their own tactics and strategies, which ultimately required them to put more trust into local allies and new equipment (such as up-armored vehicles, electronic jammers, and robots). And while this did not necessarily imply that any warfighter was safer than before, the trust helped warfighters deal internally with wartime uncertainties so that they could continue their duties and focus on mission objectives in this hostile environment.

This story illustrates how trust in people and equipment is critical for warfighters dealing with uncertainty in combat operations, particularly when making decisions that trade off security and performance. And trust will be even more critical as the U.S. military introduces more advanced military robotic systems into theatre to operate autonomously in heterogeneous teams [2]. These systems are expected to minimize the number of warfighters required to complete certain dangerous missions [58], which will help to lower the causality counts of U.S troops. But the large presence of these robots will also introduce new trust-based vulnerabilities that previously did not exist in homogeneous human teams. One of these vulnerabilities is connected to a robot’s exposure to cyber attacks, which can be used to disrupt or take over these systems for nefarious purposes.

As doctrine, the Pentagon has formally recognized cyberspace as a new domain in warfare [39], which has become just as critical to military operations as land, sea, air, and space. And the evidence that other nation-states are actively developing their own cyber attack capabilities to break into or disrupt unmanned systems can be observed in recent media reports [18] [85] [122] [123] [154]. One can expect that these cyber attack capabilities will become more sophisticated in time. In addition, one can logically extrapolate that as more unmanned systems are introduced into military operations, the value of attacking these assets will be higher. As such, our nation will be more exposed to these types of attacks and vulnerabilities in the future, which puts American warfighters and U.S national security interests in jeopardy.

Prior to 2003, the U.S. military had no fielded unmanned ground vehicles [128], despite Congress's goal in 2001 that "by 2015, one-third of operational ground combat vehicles are unmanned" [101]. But over the past decade, more than 12,000 unmanned ground vehicles have been fielded for use in Iraq and Afghanistan [128]. These first-generation systems were largely tele-operated by a single warfighter and tailored to a narrow mission [88]. But their usefulness has spurred demand for more robotic capabilities across a broader spectrum of military operations [140] [141].   Autonomous convoy operations are expected to be one of the near-term, next-generation applications of unmanned technologies [57] [116] [119]. As such, the autonomous convoy mission presents a relevant and constrained application of a computational trust problem and will be the primary application focus throughout this dissertation. The technical risk is high as it has not yet been demonstrated in any military robotics program that trust algorithms are useful in detecting and recovering from cyber warfare attacks. However, the potential payoff is high as well, since it would show the relevance and utility of incorporating computational trust concepts as a defensive capability in autonomous military systems.

It is important to also consider our specific autonomous convoy trust problem within the larger context of soft security for the next generation of advanced military robotic systems. The advanced systems we refer to will be required to operate and co-operate in highly dynamic, unstructured, and hostile environments [141], such as urban warzones, natural or man-made disaster areas, and subterranean caves and mines. These systems will also increasingly become more autonomous and more common in military operations, creating a need for robust strategic and tactical artificial intelligence [129] [140]. In particular, these military robots will need to decide to what capacity they will interact with other robots and humans, given the presence of uncertainty and partial information. In addition, cooperative multi-robot teaming applications will pose general task coupling and communication-delay challenges for these interactions. Currently, there is no working theory that takes into account all of these issues [28]. So from both a security and performance perspective, these advanced systems will need to have innate abilities to correctly interpret the behaviors of other agents within their local environment so that they can correctly decide on actions that best satisfy their individual, team, and mission objectives.

1.2 General Problem

The need for trust is not only limited to combat situations for warfighters. Trust is also vital for group unity and cooperation in more ordinary social interactions [21] [138]. Trust impacts a range of social processes between warfighters that influence the cognitive and physical strain of being at war. Often, when the trust of a warfighter becomes lower toward other people, equipment, or processes, then he will likely need to exert more effort in order to resolve perceived uncertainties. This extra effort could manifest itself into a distraction that lowers a warfighter's effectiveness at best. However, in prolonged, high-stress situations, this additional effort could also manifest itself as a persistently guarded psychological state that continuously monitors for violations of expectations and predictions.

Trust is, therefore, a critical survival tool for warfighters dealing with uncertainty, whether in combat or not. The effect of high trust in military settings has been shown to lessen the defensive monitoring of others, reduce the need for hierarchical control, improve cooperation due to increased predictability and expectations of reciprocity, improve information sharing (with less need to filter unfavorable information), lower levels of conflict (friction and dissent), and improve group performance and processes [4]. And it could be argued that similar emergent efficiencies, such as lower power consumption, faster algorithms, and the exchange of higher-quality information, have the potential to be realized in autonomous systems that use trust as a basis for dealing with uncertainty.

Conceptually, we will see that the level of trust one has toward another allows one to accept or reject assumptions about another's present states or future behaviors within a particular context. For people, cultivating trust is a natural, emotional process that factors in a lifetime of experience. For artificial agents, however, cultivating trust requires a precise mathematical description of trust and the knowledge of how to apply it to decisions and control. To this end, the general problem we address is the gap between our emotional understanding of trust and a representative, practical mathematical description of trust. This general problem is one of the key motivators for researchers working in the computational trust research domain, which our work resides in.

Computational trust research is also incorporated into a broader research area known as **network science**, which examines the interconnections among diverse networks and seeks to discover common principles, algorithms and tools that govern network behavior. The U.S. Army has shown a particular interest in advancing network science research. It has formed the Network Science Collaborative Technology Alliance (NS CTA) at the Army Research Laboratory in order to improve its ability to analyze, predict, design, and influence complex systems that interweave many kinds of networks [139]. Within the NS CTA, computational trust research is regarded as a cross-cutting research issue (CCRI) that can help enhance distributed decision-making capabilities of the Army in the context of network-centric operations (in particular, for irregular warfare and counterinsurgency) . More generally, the Trust CCRI seeks to understand the role trust plays in composite networks that consist of large systems with complex interactions between communication, information, and social/cognitive networks. As such, we deliberately formulate our theoretical work in terms of multi-agent systems and graph theory to align with the broader trust research initiatives in network science.

1.3 Research Objective

The research goal of this dissertation is to determine if computational trust can provide a feasible defensive capability against unacceptable behaviors in military autonomous systems. Thus, our research objective is to develop new or improved computational trust models, algorithms, and frameworks for trust cultivation, aggregation, and propagation in distributed teams. Our applied focus is directed toward autonomous military convoy operations due to its relevance for the U.S. Army and constrained mission profile.

1.4 Contributions

Our work contributes knowledge and techniques to the research domains of computational trust, cooperative game theory, mobile robotics, and network science. Specifically, our contributions are listed below in the order of which they appear in the dissertation.

- Developed the **confidence-doubt model** for trust-based risk determination in a relationship that conforms to the philosophical notions of trust. (Chapter Two)
- Developed an **interaction cycle** between two agents in a trusting relationship, highlighting the importance of reciprocity. (Chapter Two)
- Developed the new theory of the **cooperative trust game**, which includes a trust game class characterization and a general model. (Chapter Three)

- Proved that the highest trust payoff in a vehicle convoy occurs when the global leader is the trusted third party for all of its followers. (Chapter Three)
- Developed a new computational trust algorithm called **RoboTrust**, which calculates trustworthiness in agents by assigning the smallest value in a set of maximum-likelihood estimates based on different historical observations. (Chapter Five)
- Developed an extension to RoboTrust for the propagation and aggregation of recommendations in indirect trust computation. (Chapter Five)
- Compared the trust model performance of RoboTrust to two commonly-used probabilistic trust models: Beta Expectation and Bayes' Rule. (Chapter Five)
- Developed a distributed, discrete-time, **trust-based consensus protocol** and proved its asymptotic convergence to an agreement space. (Chapter Six)
- Analyzed the trust-based consensus protocol under two overarching conditions – static-trust and dynamic-trust – using a simple three-agent network. (Chapter Six)
- Analyzed the decentralized convoy case using the cooperative trust game theory and discovered a new way to view other vehicles within a convoy: as surrogates for the system of unobservable agents in the convoy. (Chapter Seven)

- Developed a simulated **trust-based vehicle controller** for convoy operations and analyzed its ability to detect and mitigate the bad behavior of neighboring convoy vehicles. (Chapter Seven)

1.5 Publication Productivity

This section presents the publication productivity of the author during his dissertation research period (July 2010 – July 2013). Papers with first authorship indicate material directly related to the content of this dissertation.

1. D.G. Mikulski, F.L. Lewis, E.Y. Gu, G.R. Hudas. "Trust-Based Coalition Formation in Multi-Agent Systems." *To appear in*: Journal of Defense Modeling and Simulation: Applications, Methodology, Technology. SAGE Publications. 2013.

2. D.G. Mikulski. "Cooperative Trust Games." In: Game Theory Relaunched. ISBN 979-953-307-783-2. InTech. pp. 233-250. 2013.

3. M. Aurangzeb, D.G. Mikulski, G.R. Hudas, F.L. Lewis, E.Y. Gu. "Stable Structures of Coalitions in Competitive and Altruistic Military Teams." Proc. SPIE 8741. Baltimore, MD. 2013.

4. C. DiBerardino, D.G. Mikulski, E. Mottern, T. K. van Lierop, N.J. Kott. "Large Platform Autonomy in Urban Environments." NDIA GVSETS. Troy, MI. 2012.

5. D.G. Mikulski, F.L. Lewis, E.Y. Gu, G.R. Hudas. "Trust Method for Multi-Agent Consensus. In: Proc. SPIE 8387. Baltimore, MD. 2012

6. G.R. Hudas, K.G. Vamvoudakis, D.G. Mikulski, F.L. Lewis. "Online adaptive learning for team strategies in multi-agent systems." In: Journal of Defense Modeling and Simulation: Applications, Methodology, Technology 9(1). SAGE Publications. pp 59-69. 2012.

7. D.G. Mikulski, F.L. Lewis, E.Y. Gu, G.R. Hudas. "Trust Dynamics in Multi-Agent Coalition Formation. In: Proc. SPIE 8045. Orlando, FL. 2011.

8. K.G. Vamvoudakis, D.G. Mikulski, F.L. Lewis, E.Y. Gu, G.R. Hudas. "Distributed Games for Multi-Agent Systems: Games on Communication Graphs. In: Proc. 27th Army Science Conference. 2010.

Conclusion

The motivation of this research stems not only from a desire to protect our warfighters and nation, but also from a fundamental technical gap in addressing trust-based vulnerabilities in military unmanned systems. Our work attempts to provide tools and techniques to address this specific problem while also contributing to the larger research domains of cooperative game theory, computational trust, mobile robotics, and network science.

CHAPTER TWO

TRUST AND RELATIONSHIPS

Synopsis

This chapter discusses the role of trust within interpersonal relationships. Its purpose is to establish an intuitive understanding of the trust concept that is useful to know in later chapters of this dissertation. It also reviews certain research topics related to the trust research area – in particular, knowledge, belief, and intention.

The chapter starts with Section 2.1, which explores the philosophy of trust across several important philosophical dimensions. Section 2.2, then, presents the confidence-doubt model for trust-based risk determination in a relationship that conforms to the philosophical discussion of trust in Section 2.1. Section 2.3 describes a general interaction cycle between two agents in a trusting relationship, highlighting the importance of reciprocity. Section 2.4 provides a personal case study about the author and his dog to illustrate the various cognitive concepts of trust presented in this chapter.

2.1 Philosophical Underpinnings of Trust

In this section, we examine the philosophical paradigm of interpersonal trust. Among philosophers, it is considered the dominate paradigm of trust [93] and lends itself well to our work in later chapters for multi-agent systems. Some philosophers also consider institutional trust [110], government trust [63], and self-trust [52] within the philosophical trust literature, but we will not delve into these minor paradigms.

Therefore, when we refer to trust in this dissertation, we will always implicitly assume interpersonal trust.

In general, trust helps people deal with uncertainty about others by reducing the complexity of expectations in arbitrary situations involving risk, vulnerability, and interdependence [84]. This emotional attitude is particularly useful when something cannot be gauged precisely with reasonable time or effort. The benefits of trustworthy relationships include lower defensive monitoring of others, improved cooperation, improved information sharing, and lower levels of conflict [4]. But the reliance on trust also exposes people to vulnerabilities associated with betrayal, since the motivation for trust – the need to optimistically believe that things will behave consistently – exposes individuals to potentially undesirable outcomes.

In exploring the philosophical underpinnings of trust, we will examine several important philosophical dimensions, including its nature, epistemology, value, and mental attitude.

2.1.1 The Nature of Trust

Intuitively, we understand that trust is a private, emotional attitude towards others whom we hope will be trustworthy. And for trust to be warranted in a relationship, we also intuitively know that each party in the relationship must have attitudes toward each other that are conducive to trust. But what conditions must hold for trust to be warranted in a relationship?

Philosophers generally consider the following uncontroversial conditions to warrant trust [17] [63] [91]:

- **An acceptance of risk**. Risk can be reduced by monitoring or imposing constraints, but a refusal to be vulnerable tends to undermine trust since it does not allow others to prove their own trustworthiness.
- **An inclination to expect the best**. Trust involves being optimistic that the trustee will do something for us or others; and this optimism is what makes us vulnerable. This optimism restricts the inferences one makes about the likely actions of another.
- **A belief in the competence of the other**. Trust is generally seen to be a three-part relation: A trusts B to do C. Thus, A must trust that B is competent and capable enough to do C. In addition, B must also be committed to doing C, which can be seen as a condition for trustworthiness.

Some philosophers believe that the existence and duration of commitment are sufficient properties of trustworthiness [63], while others believe that the origin of the commitment is also important since it deals with how someone is or will be motivated [93]. This is a controversial condition, however, since it is unclear what, if any, sort of motive one might expect from a trusted person.

Certain philosophers also believe that the trustworthiness can be compelled by the force of social constraints, as in the social contract view [105]. An alternative view suggests that people are motivated by their own interest to maintain the relationship they have with the truster [63]. Both of these views are instances of **risk-assessment views** of trust, which assumes that risk is low because it is in the self-interest of people to behave in a trustworthy manner [70].

A different view – **the goodwill view** – finds trustworthiness only where the trustee is motivated by goodwill [93] . Risk-assessment proponents say one can trust without presuming goodwill, but risk-assessment views (unlike the goodwill view) fail to demand that the trustworthy person care about the trust (or care about what he or she cares about). As such, the act of caring is seen as a central idea to a complete account of trustworthiness since it allows us to distinguish between trust and mere reliance.

One final view of trustworthiness considers trustworthiness as a virtue [93] – that is, a characteristic that makes one trustworthy toward everyone (not just specific relationships). This view is ideal if one thinks that the origin of a trustworthy person's commitment is important.

2.1.2 The Epistemology of Trust

The epistemology of trust concerns itself with "when" trust is warranted [37] [44] [76]. In discussing this topic, philosophers may consider whether or not it could ever be rational to trust other people. Rationality is the belief in something only if it has been verified. So, at first glance, it seems that trust and rational reflection are at odds with each other, since trust inherently involves risk, and any attempt to eliminate the risk through rational reflection could eliminate trust. Also, trust tends to make people resistant to evidence that may contradict their optimism about a trustee. So if we assume that it is rational to trust only if one has verified the other's trustworthiness, then this notion accounts for a partial trust, since it suggests that the rational truster is open to evidence that contradicts his trust. This understanding of trust is **truth-directed** or **epistemic**.

Rational trust can also be **end-directed** or **strategic** [12] [38]. Rather than verifying trustworthiness, it may be rational to trust for other reasons, such as when one has little control over a particular situation or when one wants to maintain a particular relationship.

Philosophers who agree that trust can be rational disagree about the degree to which rational reasons must be assessable to the truster. Some reasons are internally justified – that is, the reasons are based on known evidence. Other reasons could be externally justified [92] – such as body language, veiled forms of systematic oppression, or a complicated history of trusting others. These external reasons could influence the truster, even though the truster may not be aware of them.

Some philosophers provide a list of common justifiers for trust, which a trusting agent could take into account in deciding when to trust [53]. These include: the social role of the trustee, the domain in which the trust occurs, an agent-specific factor that concerns how good of a truster the agent tends to be, and the social or political climate in which the trust occurs. The last factor suggests that, while a trust relation is between two people, there may exist forces larger than those individuals, which can shape the trust toward one another.

2.1.3 The Value of Trust

According to philosophers, trust can have enormous instrumental value [49] and possibly some intrinsic value [93]. Instrumental value refers to the "goods of trust," which include:

- **Opportunities for cooperative activity**. Trust removes the incentive to check up on others (justified only in external ways), making cooperation easier.
- **Knowledge**. Philosophers writing on testimony argue that all knowledge acquisition depends on the testimony of others, since no one person has the time, intellect, and experience necessary to learn (independently) all of the facts about the world that we know.
- **Autonomy**. A skill acquired and exercised only in environments where one can trust other people to support it.
- **Self-respect and moral maturity**. Trust helps to improve the well-being of the trustee, allows one to be more respectful not only toward oneself, but also towards others.

Intrinsic value refers to the sign of respect bestowed on others (even if no goods are immediately produced). Philosophers have written relatively little about trust being worthwhile in of itself as opposed to worthwhile because of what it produces.

2.1.4 The Mental Attitude of Trust

If an agent has lost the ability to trust another due to some serious trauma, trust may not be warranted [65]. Therefore, it is important to consider how trust can be restored once it has been lost. While destroying trust is usually quick and dirty, trust creation is often slow and potentially painful [11]. The reasons for this relate to the kind of mental attitude trust is. This attitude cannot be willed, but it can be cultivated, which depends on how trust is justified.

This then begs the question: if one cannot simply decide to trust only because he wants to, is trust a type of non-voluntary belief? Philosophers disagree on the answer to this question [44] [66]. One reason for thinking that trust is not a belief is because it resembles the characteristics of emotion. These characteristics describe how emotions narrow a person's perception to evidence (such as when feeling angry at a loved one, a person may tend to focus on the things that justify the anger while ignoring to see things that make it unjustified). Similarly, if one trusts a loved one in a certain domain, he will focus on the aspects that justify the trust and ignore evidence to the contrary.

The characteristics of trust related to emotions are ones a person can try to mimic in his attitude toward others in an effort to be more trusting [93]. He can cultivate trust in others by focusing his attention on what makes them trustworthy. As such, by becoming more trusting (in a good way), a person can potentially receive the benefits of justified trust.

2.2 Confidence-Doubt Model for Trust

Having discussed the concept of trust abstractly in Section 2.1, we now present a cognitive model for trust-based risk determination in a relationship that conforms to the prior philosophical discussion of trust. This model follows the intuition behind the well-known supply and demand economics model [86]. While it lacks mathematical rigor, its purpose is to describe how trust can relate to risks associated with a truster's confidence and a trustee's perceived doubt. It concludes that the perceived risk associated with a particular context in a relationship will vary until it settles at a point where the trust expected by the trustee equals the trust level held by the truster toward the trustee. At this particular point, the truster assumes that he and the trustee assume the same level of perceived risk.

2.2.1 Components of the Confidence-Doubt Model

As the name suggests, there are two primary components to the confidence-doubt model, shown in Figure 2.1.

Confidence Curve. The confidence curve describes the amount of trust the truster is willing to have toward a trustee at a given level of perceived risk. It is positively sloped, implying a proportional relation between the level of trust and risk. This is because as trust increases, the truster generally has less incentive to monitor the trustee and more incentive to cooperate and share information with the trustee, exposing the truster to a higher risk for betrayal. Factors that influence the confidence curve may include a truster's optimism for reciprocity and external forces such as social or political climate.

- **Doubt Curve**. The doubt curve describes the amount of trust expected by the trustee from the truster at a given level of perceived risk. It is important to note that the expectation of the trustee is viewed from the perspective of the truster. The curve is negatively sloped, implying an inversely proportional relation between the level of trust and risk. This relation directly addresses how trust can be undermined when a truster refuses to accept a sufficient level of vulnerability, casting doubt and increasing the risk that the trustee might reject a relationship with the truster.

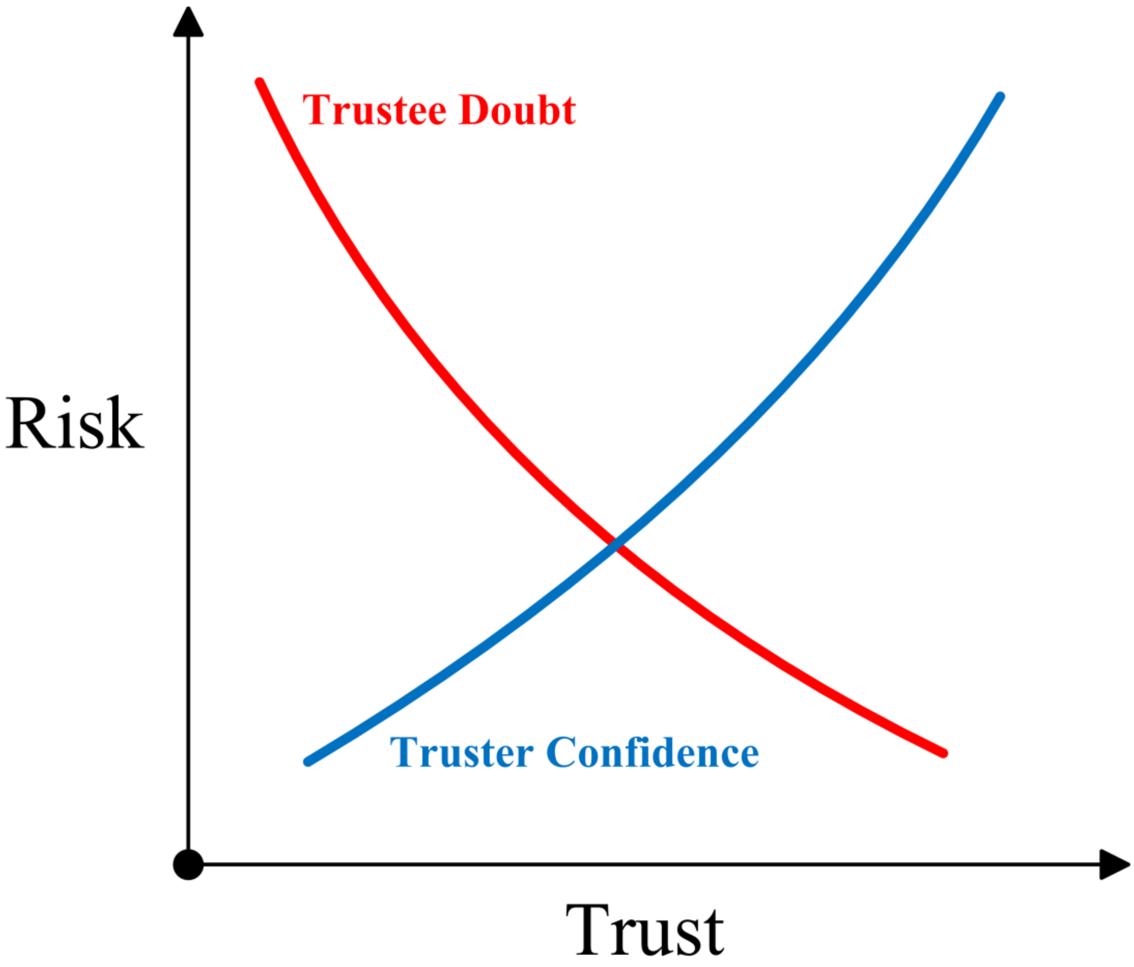


Figure 2.1. Confidence-Doubt Model for Trust

2.2.2 Risk Equilibrium

The risk equilibrium is defined as the trust-risk pair where the trust expected by the trustee is equal to the trust held by truster, represented by the intersection of the confidence and doubt curves (Figure 2.2). Risk that is accepted by the truster and is higher than the equilibrium point implies that the truster is more vulnerable to betrayal than the trustee (i.e. too trusting), and cognitively seeks to lower his risk. On the other hand, risk accepted by the truster lower than the equilibrium point implies that the truster is guarded in his relationship within the trustee (i.e. not trusting the trustee enough), and expects the trustee to take on more of the perceived risks than the truster.

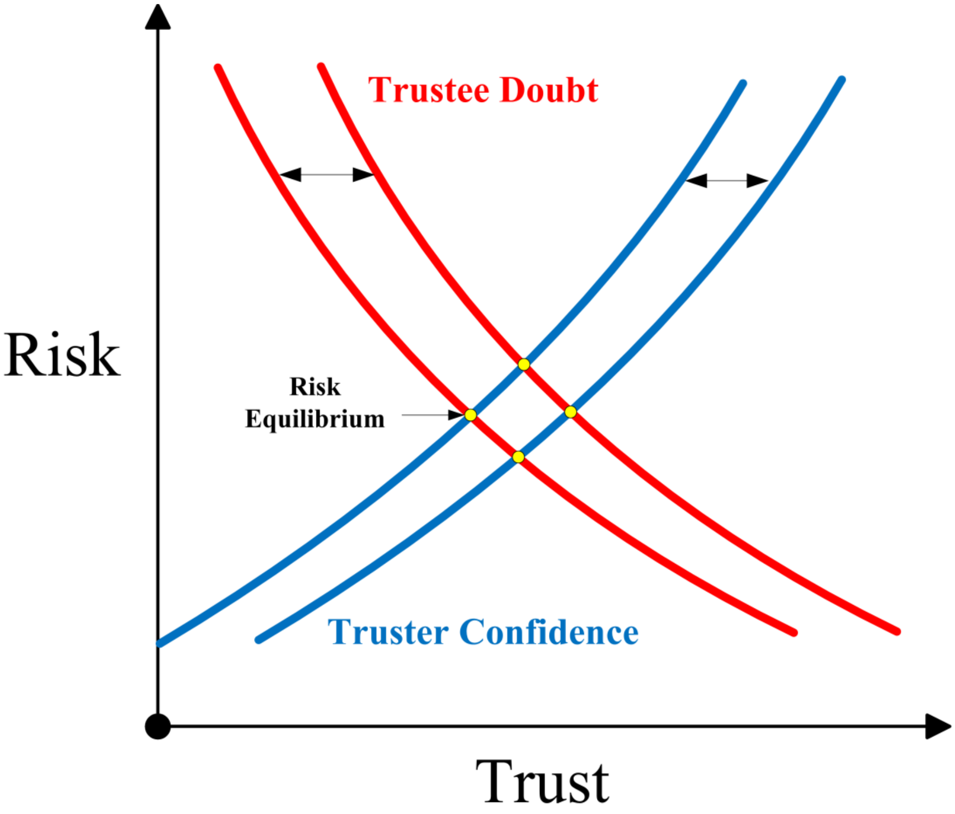


Figure 2.2. Risk Equilibrium Points in the Confidence-Doubt Model.

The equilibrium point can change if either the confidence or doubt curves shift left or right due to non-risk determinant factors. This is different than movement along the curves where trust and risk both change at the same time. For example, an increase (right-shift) in confidence suggests that the truster is willing to hold more trust towards the trustee at a given risk than he did before. Similarly, an increase (right-shift) in doubt suggests that the trustee requires more trust from the truster at a given risk than he did before.

The effects on the equilibrium point due to a curve shift can be summarized by the following rules:

1. If the doubt increases (right-shifts) and the confidence remains unchanged, then the level of trust from the truster is too low. This leads to a higher risk equilibrium point.
2. If the doubt decreases (left-shifts) and the confidence remains unchanged, then the level of trust from the truster is too high. This leads to a lower risk equilibrium point.
3. If the doubt remains unchanged and the confidence increases (right-shifts), then the level of trust towards the trustee is too low. This leads to a lower risk equilibrium point.
4. If the doubt remains unchanged and the confidence decreases (left-shifts), then the level of trust towards the trustee is too high. This leads to a higher risk equilibrium point.

2.3 Interaction Cycle between Two Trusting Agents

In Section 2.2, we depicted the relationship between trust and risk for both the truster and trustee. However, this model only described a uni-directional, "give-and-take" from the perspective of the truster. A healthy trusting relationship, by nature, is bi-directional between two agents, where both agents have to give-and-take in order for the relationship to remain stable for an extended period of time. The idea of reciprocity – a core expectation in a trusting relationship – was implied in Section 2.2, but explicitly absent in the cognitive model.

In this section, we fill in the reciprocity gap by describing a general interaction cycle between two agents in a trusting relationship. Section 2.3.1 discusses the general motivation agents have to establish a trusting relationship while Section 2.3.2 provides detailed descriptions of each interaction element in the two-agent interaction cycle.

2.3.1 The Motivation to Interact with Others

The motivation for an agent to interact with another agent usually stems from the existence of an unresolved problem that the agent wants to resolve, but believes it cannot without the help of the other agent. After all, if an agent believes it could resolve a problem without the assistance of another, we would assume that it would. Therefore, we see that the motivation to interact with others is a selfish motivation resulting from a deficiency of capabilities, knowledge, or resources. An agent who is motivated to interact with another agent, thus, intends to take advantage of the other agent's capabilities, knowledge, or resources in order to resolve its own personal problems. Having said this, in order to maintain the interaction, the agent must be willing to reciprocate by sharing its own capabilities, knowledge, or resources with the other agent since the other agent likely has the same sort of motivation.

Before two agents begin to resolve an arbitrary problem, the agents must be able to describe the problem clearly so that it can be verified as solved (or partially-solved) by all parties. However, there is no standard way to describe arbitrary problems. As such, so that we may continue discussing the notion of problem-solving in multi-agent systems, we must now establish a new, more precise way of describing general problems. For the purposes of this work, we define a problem as "a series of actions not yet taken towards an objective."

This simple definition of "problem" transforms the vague notion of a problem in terms of actions. It suggests that a problem can only exist if an agent exists and the actions necessary to resolve an existing problem have not been done. An action can be anything that some agent can do to the broadest extent possible, and the process of solving a problem is simply the transition from action to action until the final action toward the objective is complete. As such, a problem is considered solved (non-existent) if there are no further actions to take.

When describing a problem, each agent must know their current state and have, at least, some vague idea of a final state. As such, a problem, at its most vague level, can be described as an agent's single action of changing from its current state to the final state. However, in most cases, an agent cannot simply change states instantaneously. An agent is limited by the set of its potential actions, of which only a subset of these may be available at any given time. Using any available actions, an agent can transition from one state to another state, constantly monitoring whether or not it has reached the final state. However, there is no guarantee that an agent will ever take (nor has the ability to take) all the necessary actions to solve an arbitrary problem. Furthermore, inaction does not imply that a problem will remain static, since often, problems are dynamic in nature due to external influences. Hence, to truly solve a problem, an agent must not only traverse a series of actions to the final state, but also maintain the final state in spite of any external influences. So if the complexity and scope of a problem is sufficiently high, then it may advantageous to approach a solution from a multi-agent perspective. This is because simple interactions between multiple agents can generate complex emergent behaviors at the macro-level [80] [106], and thus, this serves as a sufficient motive to interact with other agents.

2.3.2 The Elements of Interaction

Having established the general motivation for "why" two agents may interact with each other, we now provide a general process of "how" we consider two agents to directly interact with each other. This process describes the flow of information between distinct high-level elements of interaction, shown in Figure 2.3. The flow of information passes through the abstract zones of common and private knowledge, indicating what information is observable and potentially accessible to all, or hidden and accessible only to one agent, respectively. There are two loops in the interaction cycle – an inner loop in which a single agent updates its knowledge on the basis of its own actions; and an outer loop in which two agents update their knowledge on the basis of observed results from each other.

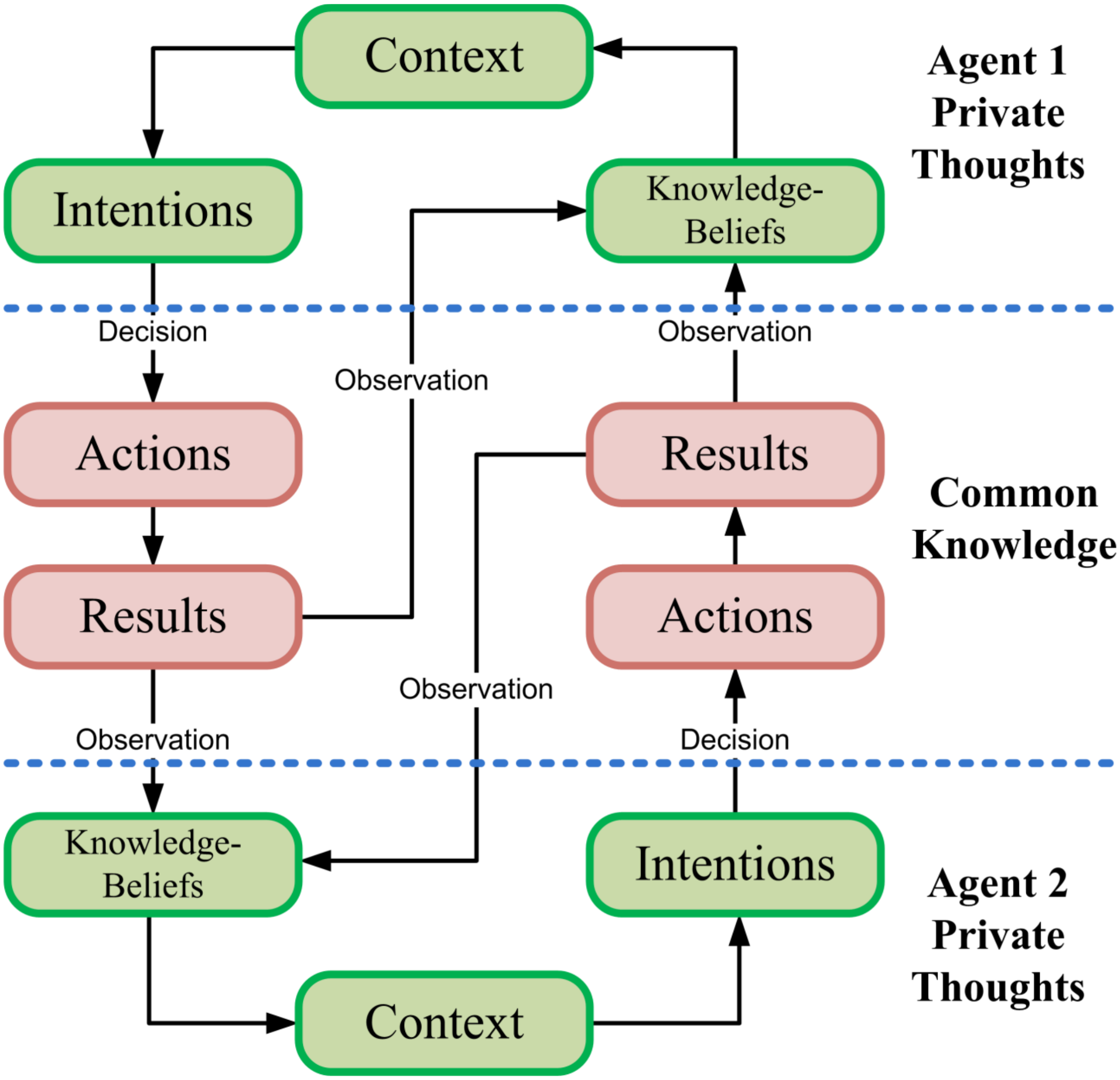


Figure 2.3. The Interaction Cycle between Two Trusting Agents.

The following subsections provide a detailed description of each interaction element in the cycle.

2.3.2.1 Knowledge-Belief Element

The knowledge-belief element stores and manages an agent's knowledge and beliefs, which are based on direct or indirect observations of the environment and agents within it. While knowledge is presumed to always be true, beliefs need not necessarily be realistic or accurate. As such, this element must consider addressing the problem of belief revision – the process of revising an existing state of belief on the basis of newly learned information.

Some theoretical work has been done with regards to reasoning about knowledge and belief in distributed systems in a more precise manner [125]. One way to reason about knowledge is through an **n-agent partition model** $\mathcal{M}$ over a language $\mathbb{L}$, which denotes the sets of possible worlds ($\mathcal{I}_i$) that are equivalent from the point of view of an agent i. Worlds ($\mathcal{W}$), within this framework, are used to specify the concrete state of affairs of some situation or environment. This partition model also includes an interpretation function that determines which sentences in the languages are true in which worlds (i.e. $\Pi: \mathbb{L} \rightarrow 2^{\mathcal{W}}$). To help define when a statement is true, the partition model also defines a logical entailment. Using the notation $K_i\varphi$ as "agent i knows that φ" and $w \in \mathcal{W}$, the logical entailment $\Longrightarrow$ is defined as:

- For any $\varphi \in \mathbb{L}$, we say that $\mathcal{M}, w \Longrightarrow \varphi$ if and only if $w \Longrightarrow \Pi(\varphi)$
- $\mathcal{M}, w \Longrightarrow K_i\varphi$ if and only if for all worlds w', if $w' \in \mathcal{I}_i(w)$, then $\mathcal{M}, w' \Longrightarrow \varphi$

The second part of the logical entailment definition states that we can only conclude that agent i knows φ when φ is true in all possible worlds that i considers indistinguishable from the true world.

While partition models allow one to rigorously reason about knowledge, they end up being cumbersome when models become large. In such cases, it may be possible to reason about such models using an **axiomatic system based on modal logic**. In this context, a modality represents a particular type of judgment regarding a sentence. For example, the modal operator $\square$ means "necessarily" (and thus $\square\varphi$ is read as "φ is necessarily true"). Similarly, the modal operator $\lozenge$ means "possibly" (and thus, $\lozenge\ \varphi$ is read as "φ is possibly true"). The semantics are defined in terms of possible-worlds structures, also known as Kripke structures [77]. One can think of Kripke structures as directed graphs, with the nodes being the classical propositional models and the arcs representing accessibility (or binary relation) on these models.

The axiomatic theory of the partition model establishes axioms and constraints on the accessibility relation. The following summarizes these axioms, known as KDT45. One should note that axiom D can be derived from K and T, so the system is more commonly referred to as KT45. The reason this axiomatic theory captures the properties of knowledge in the partition model is because the equivalence relation is reflexive, transitive, and Euclidean.

- **Axiom K**. $(K_i\varphi \land K_i(\varphi \to \psi)) \to K_i\psi$. States that an agent knows all of the tautological consequences of some knowledge (omniscience).
- **Axiom D**. $\neg K_i(\varphi \land \neg\varphi)$. States that an agent cannot know a contradiction (consistency). The accessibility relation is *serial*.

- **Axiom T**. $K_i\varphi \rightarrow \varphi$. States that it is impossible for an agent to know something that is not true (veridity). The accessibility relation is *reflexive*.
- **Axiom 4**. $K_i\varphi \rightarrow K_iK_i\varphi$. States that when an agent knows something, it knows that it knows it (positive introspection). The accessibility relation is *transitive*.
- **Axiom 5**. $\neg K_i\varphi \rightarrow K_i\neg K_i\varphi$. States that if an agent does not know something, then it knows that it does not know it (negative introspection). The accessibility relation is *Euclidean*.

The concept of belief can also be discussed in terms of KD45 axiomatic theory [60]. Essentially, all the "knows" (K_i) above are replaced with "believes" (B_i). The semantics of this logic are Kripke structures with accessibility relations that are serial, transitive, and Euclidean. Furthermore, knowledge and belief can be combined by merging both semantic structures of knowledge and belief and having two sets of accessibility relations over the possible worlds. Knowledge becomes KD4 and belief remains KD45. However, such a merge does not capture any interaction between knowledge and belief. As such, a third logic describing these interactions is necessary within the context of a KB-structure.

- $K_i\varphi \rightarrow B_i\varphi$. States that if an agent knows something to be true, then he also believes it is true.
- $B_i\varphi \rightarrow B_iK_i\varphi$. States that if an agent believes something, then he also believes that he knows it.

- $B_i\varphi \rightarrow K_iB_i\varphi$. States that if an agent believes something, then he also knows that he believes it.
- $\neg B_i\varphi \rightarrow K_i\neg B_i\varphi$. States that if an agent does not believe something, then he knows that he does not believe it.

Extensions of the axiomatic theory have added a quantitative component (probability) to knowledge and belief statements, allowing for an agent to express to what degree it knows or believes a particular proposition [124]. A straightforward way is to take a partition model and overlay a commonly known probability distribution (called the "common prior") over the possible worlds. This allows one to quantify how likely an agent considers each possible world. However, this type of approach is seriously constrained as it implicitly assumes that the partition structure for each agent is common knowledge and that the beliefs of the agents are based on a common prior. This means that the beliefs of an agent are the same within all worlds of any given partition. While it is possible to give agents probabilistic beliefs without assuming a common prior, it brings up various complexities outside of the scope of this subsection.

Having described a number of ways to reason about knowledge and beliefs, we now consider the problem of **belief revision** [107]. In general, when new information (φ) is consistent with old beliefs, one simply adds the new information to the old beliefs and then takes the logical closure of the union. For the probabilistic case where the prior beliefs are in the form a probability distribution $P(\cdot)$, we use the posterior distribution $P(\cdot\,|\varphi)$.

When new information is inconsistent with old beliefs, one can process belief revision in a KB-model by removing all worlds that are inconsistent with the new information. Thus, the KB-model is reduced to a new model that is consistent with old beliefs and the new information. This form of belief revision is characterized in two ways: with AGM postulates and axiomatic theory. The AGM postulates, named after the names of their proponents (Alchourrón, Gärdenfors, and Makinson), are properties that a revision operator should satisfy in order to be considered rational [7]. Axiomatic theory of belief revision uses a non-monotonic consequence relation to define properties that satisfy the rational consequence relation [20]. Probabilistic belief revision [15] is an extension of axiomatic belief revision that uses nonstandard probabilities within a lexicographic probability system in order to address the issue of conditional belief $P(A|B)$ when $P(B) = 0$.

Before concluding this subsection, we will briefly mention other forms of belief dynamics found in the literature [50] [75] [89].

- **Belief Expansion**. The addition of a belief, regardless of whether or not it leads to a contradiction.
- **Belief Contraction**. The operation of removing just enough from a theory to make it consistent with new evidence.
- **Belief Update**. While similar to belief revision, it is subtly different. Belief revision assumes that new evidence supports facts that were true all along. Belief Update does not make this assumption and changes the old beliefs to support the new evidence. For example, if an agent believes it is not raining, but suddenly feels raindrops, then belief revision would assume that his original belief was wrong. But in the case of belief update, the assumption is that he was right up until the new evidence of feeling raindrops.

- **Belief Arbitration**. An egalitarian approach that replaces the prioritization rule from the AGM postulates with a fairness axiom. The intuition is that when new evidence is inconsistent with old beliefs, then each side must give up something to resolve the inconsistency.
- **Belief Fusion**. The process of merging two belief states. A belief state is the total preorder on possible worlds that describes conditional beliefs (current beliefs as well as hypothetical beliefs). The basic conflict of merging belief states is not simply inconsistency between two beliefs, but rather inconsistency between the two orderings on possible worlds. To resolve this conflict, it has been suggested that agents should place a strict "credibility" ranking on the belief sources and accept the highest-ranked opinion offered on every pair of worlds.

2.3.2.2 Context Element

The purpose of the context element is to frame an agent's knowledge and beliefs in an appropriate context and deal with any uncertainties associated with the context. Contextualizing is critical in this stage as it involves placing interrelated conditions on knowledge and beliefs that influence the meaning of what has occurred or is occurring in the environment or with the agents. Without these conditions, facts and opinions could be interpreted arbitrarily due to potential ambiguities.

To illustrate the importance of context, let us consider a simple example. Let us assume the following fact to be true: that a co-worker requires help to fix a leaky faucet. In analyzing this fact, we might reasonably conclude that someone who works in our company does not have the skill set necessary to correct a continuous drip from a faucet. Our conclusion is based on a context that those who need help with plumbing do not know how to plumb. But now, suppose it is common knowledge that we work for a plumbing company. How does this change the meaning of the original fact? Perhaps we might conclude that a co-worker is overloaded with plumbing jobs. Or perhaps a co-worker has an unusually difficult faucet to repair. Perhaps our original conclusion is still accurate if our co-worker is an entry-level apprentice. Or perhaps the true conclusion is some combination of the previous conclusions. It is clear that there is some ambiguity to the meaning of the known facts. However, in order to discover the true meaning of the facts, a context needs to be selected and evaluated against additional facts or opinions. As such, the selection of the context will ultimately guide the manner of which the discovery process evolves.

To deal with the uncertainties associated with a selected context, an agent must consider the potential payoffs and risks associated with the context as well as its trust towards the other agent with respect to this context. An agent who is able to overcome the uncertainties will depend on the trust to maintain an interactive relationship. This does not imply that the agent is less vulnerable to the risks or more likely to receive a payoff because of this trust. It only suggests that the agent is able to deal internally with the uncertainties in a manner that allows it to continue its interactions with the other agent. The level of trust, however, will influence the intentions of the agent as well as of the level of desired interactivity with the other agent.

2.3.2.3 Intention Element

The intention element represents an agent's thoughtful and deliberate planning of actions aimed at reaching a particular end goal. In other words, the agent uses this element to describe the problems which it will try to resolve, as discussed in Section 2.3.1.

To deal with the notion of intention, we must also deal with the notions of commitment, capability, desires, and goals [33]. Listed below are some intuitions about these notions.

- Desires are unconstrained and do not need to be achievable or consistent. Goals must be consistent and believed to be achievable. Intentions are like goals, but in addition must persist in time. Thus, intentions imply goals, and goals imply desires – these mutual constraints are sometimes called rational balance.
- Intentions and goals are both future-directed. Intentions come in two varieties – intentions to achieve a particular state and intentions to take a particular action.
- Plans are a set of intentions and goals, and in general, are "partial" – that is, the intentions and goals may not be directly achievable by the agent.
- Plans may produce additional goals or intentions, but since plans must be internally consistent, the plan constrains the addition of new goals and intentions.
- Intentions are persistent, but not irrevocable. Also, agents do not need to intend any anticipated side effects of their intentions.

There has been some work to capture the notion of intention formally through dynamic logic, which uses modal logic with explicit "motivational" modal operators [148]. However, this is only a sketch of a formal theory since modeling intention is considered more complex, messy, and controversial than modeling knowledge and belief. The theoretical sketch uses two primitive modal operators – belief (B) and goal (G) - to define intention. These operators are both intended to be interpreted via possible-world semantics. B is a standard KD45 belief operator and G has no restriction other than it must be serial. However, the G accessibility relation must be a subset of the B accessibility relation, since goals must be consistent with beliefs. With B and G, it is possible to define the concept of an "achievement goal" – a goal that has yet to be achieved. To model commitment, the notion of a "persistent goal" is defined – an achievement goal that an agent will not give up until it believes that it is true or will never be true. And finally, to model an "intending action", the agent must have a persistent goal that it believes it will take an action towards – and then actually take the action.

2.3.2.4 Action Element

The action element describes the actuation of an agent in its environment. This requires the agent to use some source of energy and convert it into some kind of activity, such as motion or communication. We assume that any action taken by an agent is determined by the mental state that precedes it (whether conscious or unconscious); hence we consider actions to be causally deterministic from a decision. Inaction – the act of doing nothing – is therefore interpreted as an inability to make a decision or the lack of a capability to perform a desired action, and is thus not considered a valid action.

2.3.2.5 Result Element

The results element describes the consequences of actions or environmental changes. These consequences can be either intentional or unintentional; but the overall impact of the results depends on the point of view of an agent and its own goals.

The result element assumes a causal relationship between actions and results. We state this explicitly because, historically, this assumption has not been universally held. For example, the classical thinking of Aristotle and Sir Francis Bacon supported the idea that causality is the grounding for all knowledge [45] [62]. Even Galileo and Sir Isaac Newton never denied causality in their scientific research - however, they seemed to have compartmentalized it, ignored it, and then moved on to formulate their ideas [41]. This can be observed in their research on gravity – they never sought to explain the causes of gravity, but rather worked to formulate the mathematical description that can predict its behavior [23] [47]. In the 18th century, however, David Hume was the first to reject causality as a requirement for knowledge [67]. He argued that repetition may lead to increased expectations, but that this does not imply some deeper causal relationship. Hume's analysis suggests that science and knowledge should not deal with certainty (unconditional and invariable sequences) via deterministic logic. Instead, he believed that science and knowledge should be grounded in probability theory.

With all this said, people seem to have a deeply rooted desire to characterize knowledge into causal relationships, whether correctly or incorrectly. And this desire may hint to the reason why trust exists and is leveraged so frequently in decision making. Whether accurate or not, trust tends to simplify the knowledge extracted from observed results, and this simplification provides a stable enough basis to proceed forward with (or totally abandon) cooperation and collaboration with another agent.

2.4 Case Study: Trust between the Author and His Dog

In this final section, we provide a personal case study to illustrate the various concepts and models of trust presented in this chapter. This case study is a first-person account of an experience between the author and his dog, Abby, over several weeks during the "housebreaking" process. This experience not only served as one of the initial inspirations that launched the author on the trust research path for this dissertation, but also highlighted for him the universality of the trust concept between heterogeneous intelligent agents. The experience allowed him to consider the possibility of incorporating trust concepts into intelligent machines and how that capability might foster the potential for richer and more meaningful interactions between humans and robots.

2.4.1 The Story

As a puppy, Abby was causing our family a serious inconvenience with "accidents" around the house. So to solve this problem, I decided to housebreak her by training her to ring a small desk bell next to the front door. At the beginning of the training, it was relatively simple for Abby to understand the concept between pressing the bell and going outside. However, her understanding of the bell was unfortunately limited since she would sometimes ring it to go outside to play.

To correct this issue, Abby's context of the bell needed to be reshaped to match my intended meaning. However, without a rich language to communicate this new context to Abby (as you might with another human), I had to help her discover it via direct physical interactions. This meant that instead of simply responding to every bell ring by taking her outside, I had to guess Abby's intent for ringing the bell from context clues surrounding her behaviors, and then decide whether or not to take her out. A false positive would result in her playing while I waited. A false negative would result in her having an accident inside the house.

For better or worse, my change in behavior resulted in more uncertainty for Abby. She could no longer completely trust that I would take her outside every time. Whenever I ignored her bell rings, it amounted to a betrayal from Abby's perspective (regardless of her intentions), which consequently produced fewer bell rings on average and more accidents inside the house.   It appeared as if Abby was regressing in her training.

But in reality, she was simply confused about the meaning of the bell, since my behaviors no longer matched her expectations. She needed more precise feedback on what I considered acceptable and unacceptable. As such, I made the following adjustments to my behaviors: accidents without a bell ring would result in a reprimand while accidents with a bell ring would result in a small amount of praise for trying. And to discourage Abby from ringing the bell to play, I purposely limited the amount time spent outside so that the incentive to play was minimized. If she correctly rang the bell and relieved herself, then she would receive a treat to indicate a strong amount of praise.

With this type of feedback, I started to feel that Abby's understanding of the bell improved over time. She became accustomed to the pattern of being allowed outside after feedings and naps. Also, she tended to wait for me at the door longer after a bell ring than before, indicating a high level of trust that I would eventually let her out. All this, interestingly, resulted in a surprising consequence, however. I noticed that the better she understood my intended meaning of bell, the more difficult it was for me to discern her false alarms. In fact, on occasion, I would incorrectly interpret a legitimate ring, even after feeling confident in her correct understanding of the bell. It was at this point that it occurred to me that I needed to reciprocate the trust Abby was loyally giving me for some time, and to allow her outside any time she rang the bell – regardless of whether or not I thought it was a legitimate signal. This decision effectively eliminated all accidents inside the house without significant impact to the false positives.

Abby is now currently fully housebroken. Having said this, she still occasionally takes advantage of my trust and rings the bell to play outside. So while it may seem contradictory to conclude that she is in fact fully housebroken, on personal reflection, I realized that spending time with a playful dog isn't the most terrible thing in the world. So in some ways, Abby may have helped me to discover a deeper meaning behind the bell ring than the one I originally considered.

2.4.2 Discussion

In this discussion, we will use the confidence-doubt model to describe the relative changes in perceived risk between Abby and me during the housebreaking process. As Abby's master, I set the parameters of the relationship, which were not subject to negotiation. And yet, a type of negotiation needed to occur due the dependencies of our relationship for this context. Abby depended on me to open the door to let her out, while I depended on Abby to give me an accurate signal.

Initially, Abby needed to understand the causal relationship between the bell ring and act of going outside. As the story alluded to, establishing this understanding in Abby was relatively easy. Thus, the main problem concerned itself with *when* a bell ring was acceptable. In Abby's mind, a bell ring equated to her desire to go outside for any reason. The goal, however, was for a bell ring to equate to Abby's desire to go outside for only one reason – to relieve herself.

The problem began when I began to lose confidence that Abby understood my intended meaning of the bell, despite my best intentions to train her properly (Figure 2.4). She ended up spending too much time sniffing the grass and playing with the twigs, without providing the expected results. Because of this, I began the ignore some of the bell rings, thinking it would be easier for me to learn how to guess which rings are legitimate. And as a result, Abby started to believe that I doubted her signals and did not know why I suddenly changed my behaviors (Figure 2.5). She, after all, was doing exactly what she was trained to do. Her confidence in using the bell plummeted, and her perceived risk to using the bell grew much higher than it used to be. Because of this, Abby began to ring the bell less often, resulting in an increase in

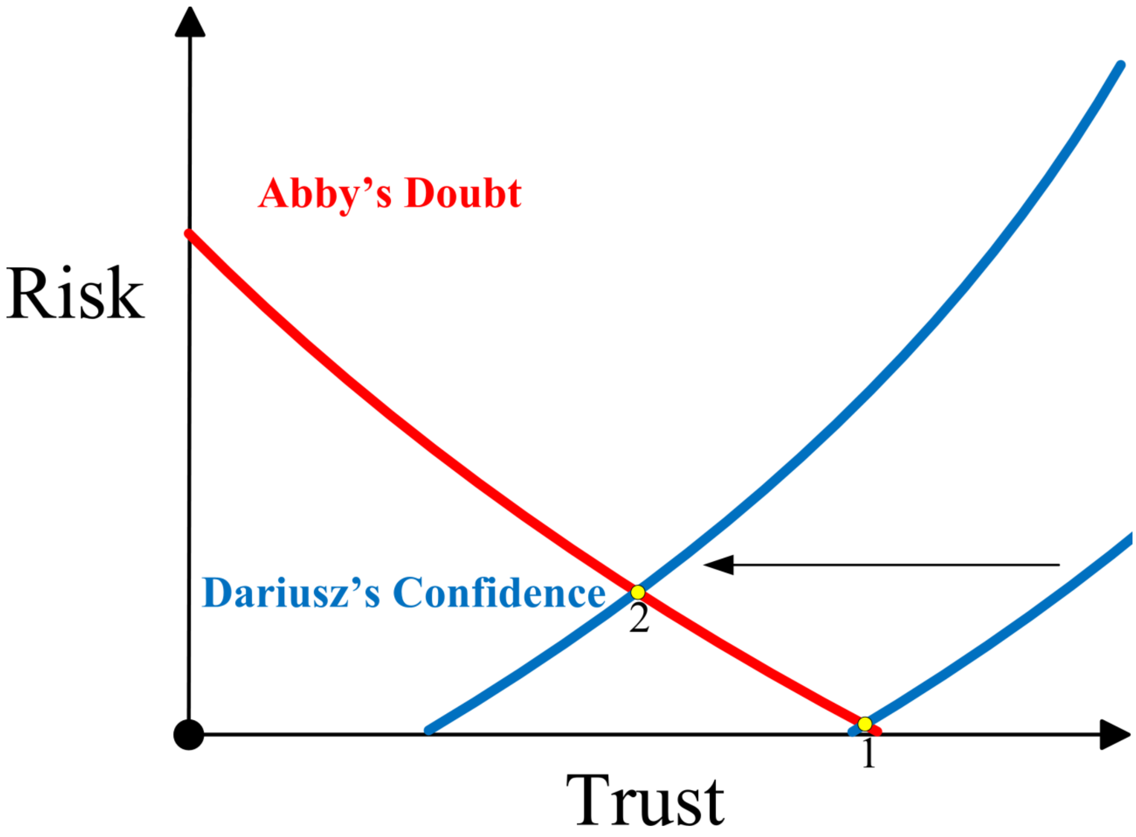


Figure 2.4. Dariusz Loses Confidence in Abby’s Behavior. Confidence decreases when Dariusz wait too long without Abby’s expected results, causing the risk equilibrium to move from point 1 to point 2. The was no change in Abby’s perceived doubt since she was not aware of any changes to Dariusz’s context.

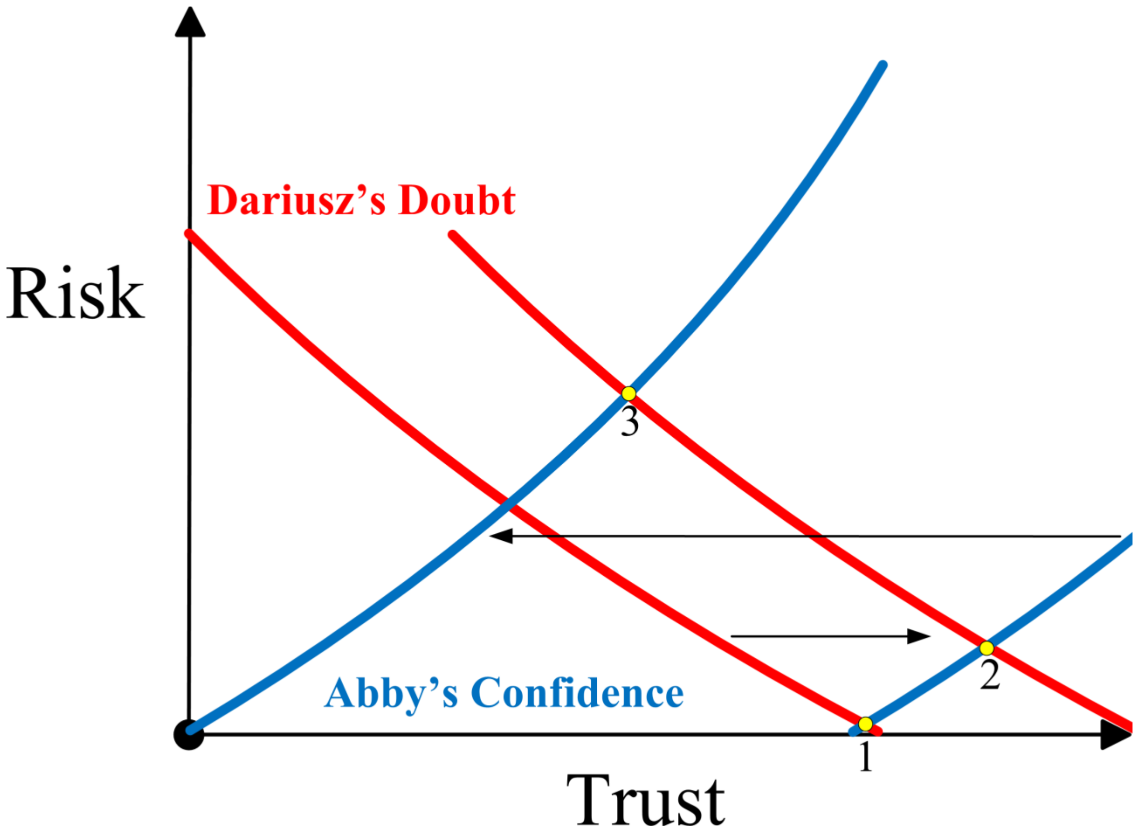


Figure 2.5. Abby Loses Confidence Due To Bell Confusion. Abby believes that Dariusz’s doubt increases when he ignores her bell rings, causing the risk equilibrium to move from point1 to point 2. In addition, Abby’s confusion causes her confidence in Dariusz to decrease, resulting in a risk equilibrium move from point 2 to point 3.

stealthy accidents inside the house and higher perceived risks from me (Figure 2.6). Clearly, my faulty strategy produced a significant amount of uncertainty for Abby, and she was searching for a way to deal with it. Abby needed more information about my intentions regarding the bell than I was actually providing. This meant that I needed to use her mistakes as opportunities to guide her towards my intentions. So each time she made a mistake, I provided her with appropriate feedback to guide her towards my desired goal. This process lasted for weeks, and tested my patience, resolve, and carpets. But over time, Abby started to learn to associate the bell ring to her full bladder, and thus, her confidence in using the bell improved (Figure 2.7). I was able to observe this increase in confidence by the predictability of her bell rings during different times of the day and her patience when she quietly sat next to the bell after ringing it. From my point of view, her doubt was decreasing (Figure 2.8). At the same time, my confidence was improving because of Abby's high success rate. In fact, her performance was so good that I felt challenged in discerning between legitimate and illegitimate bell rings. It was at this point that I chose to take a leap of faith and stop trying to guess Abby's intentions. I decided to simply trust that she actually intended to answer the call of nature whenever she signaled. And as it turned out, this was the ultimate signal Abby was looking for from me, which she discovered as her confidence in my responses to the bell improved (Figure 2.9).

Trust interactions like this are not unusual in relationships, particularly when expectations are different between agents. In general, we see increased uncertainty, which can lead to higher perceived risks at a given trust level. However, if there is a willingness to negotiate, then trust can be cultivated and perceived risk can be managed to an appropriate level for cooperation. In some sense, both parties need to accept the risk of a betrayal at some point in the future in order to cultivate the trust that it will not occur. This may be the only certainty in a trusting relationship.

Conclusion

To summarize, this chapter presented trust within an interpersonal relationship between two agents. In addition to reviewing the philosophical underpinnings of trust, it provided high-level models for trust-based risk determination and bi-directional interactivity. It concluded with a case study of a personal account between the author and his dog, illustrating the universality of the trust concept in heterogeneous intelligent agents. This particular case study allows us to consider the possibility of incorporating trust concepts in intelligent machines and imagine how such a capability could result in rich, meaningful interactions between humans and robots.

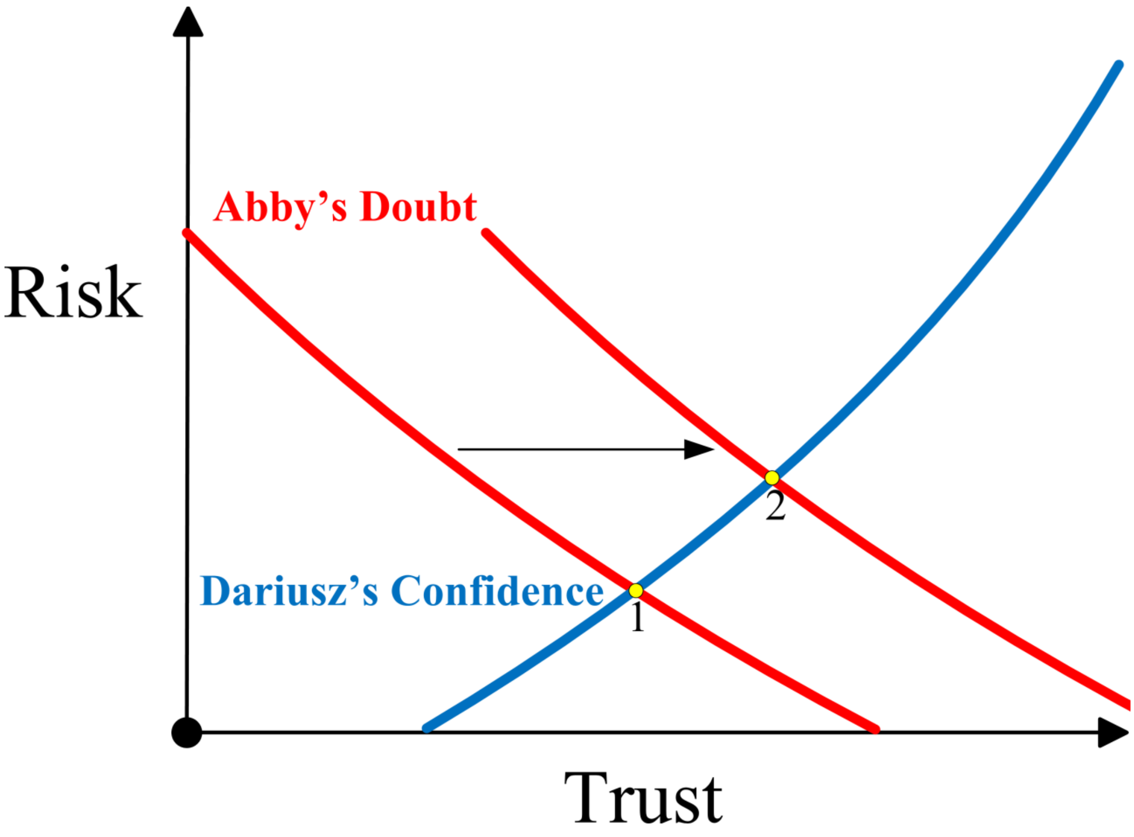


Figure 2.6. Dariusz Perceives Increase in Abby's Doubt. Since Abby rings the bell less has more accident, Dariusz interprets this as an increase in Abby's doubt. This results in a risk equilibrium move from point 1 to point 2. There was no change in Dariusz's confidence towards Abby since he already knows that Abby does not understand the bell.

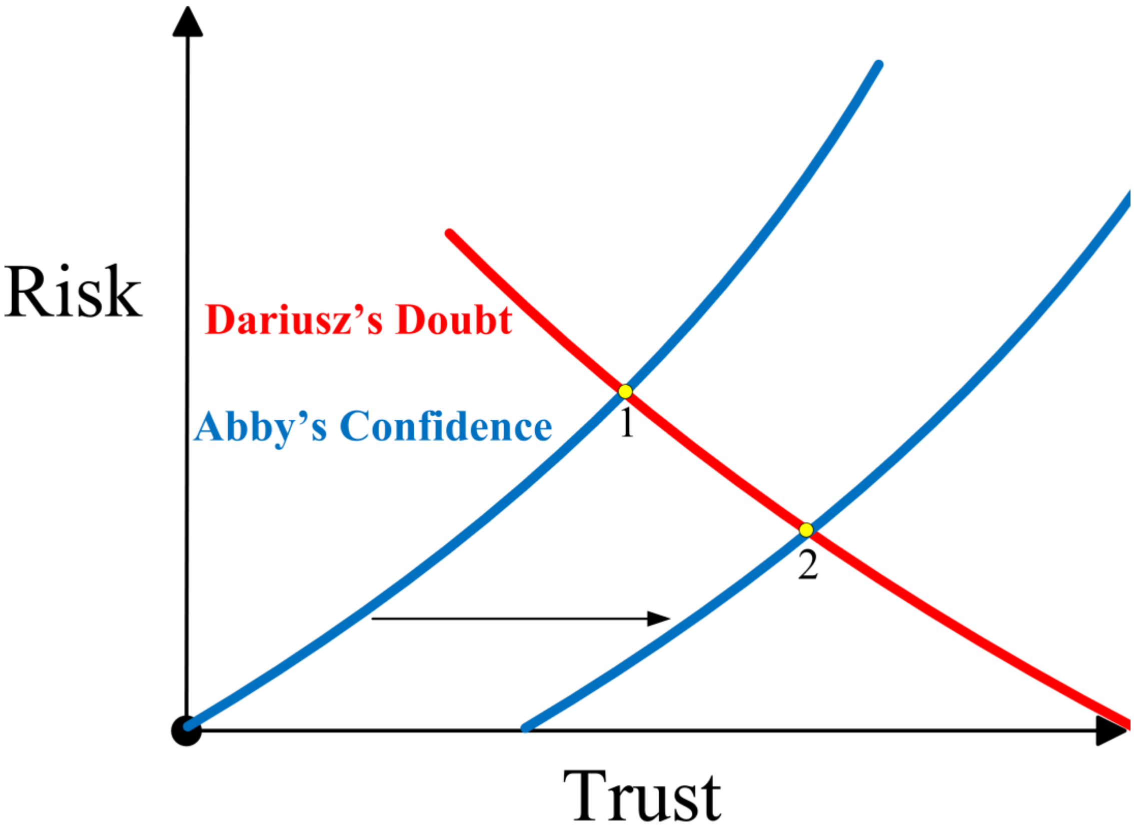


Figure 2.7. Abby's Confidence towards Bell Increases. Because Abby is able to comprehend the meaning of Dariusz's feedback, her confidence increases, causing the risk equilibrium to move from point 1 to point 2. Dariusz's perceived doubt remains unchanged since he continues to ignore some of the bell rings.

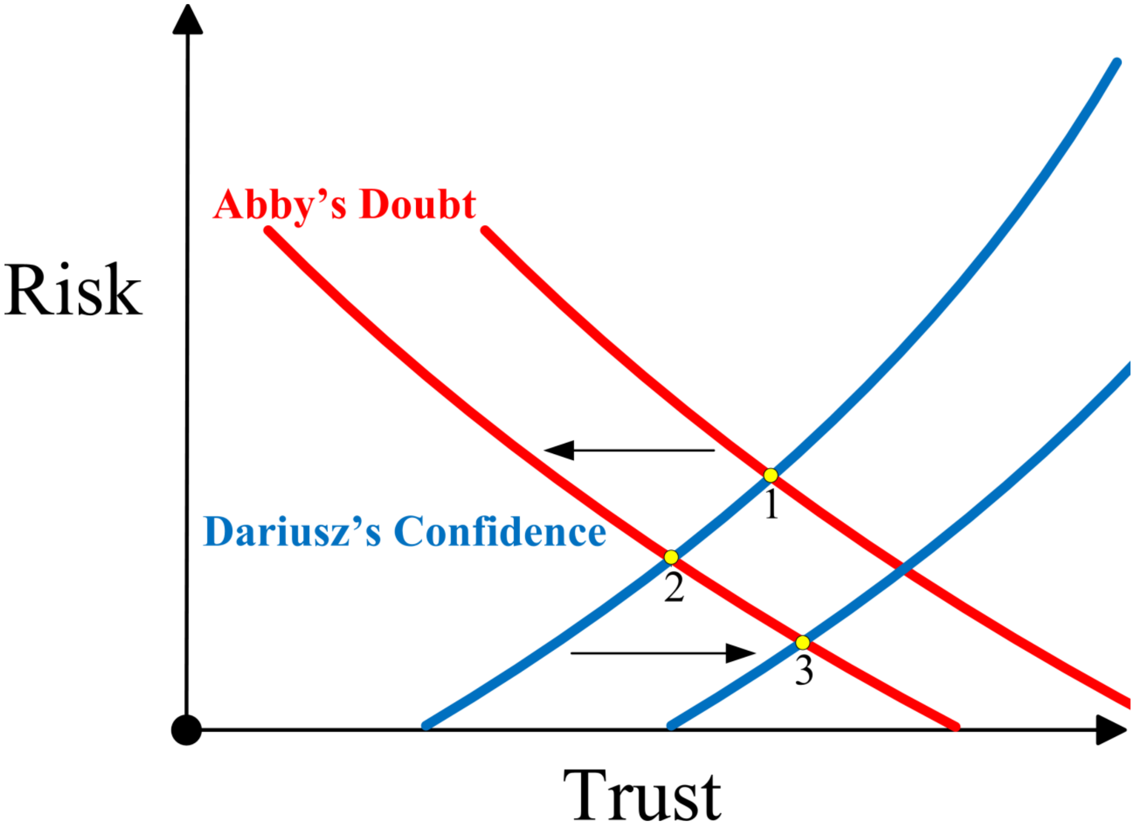


Figure 2.8. Dariusz Perceives Decrease in Abby's Doubt. Since Abby is reliably ringing the bell after feedings and naps, and waiting longer at the door after a bell ring, Dariusz interprets this as a decrease in Abby's doubt, causing a risk equilibrium move from point 1 to point 2. Abby's high success rate, along with an increased difficulty in detecting false positives, causes Dariusz's confidence to increase, resulting in a risk equilibrium move from point 2 to point 3.

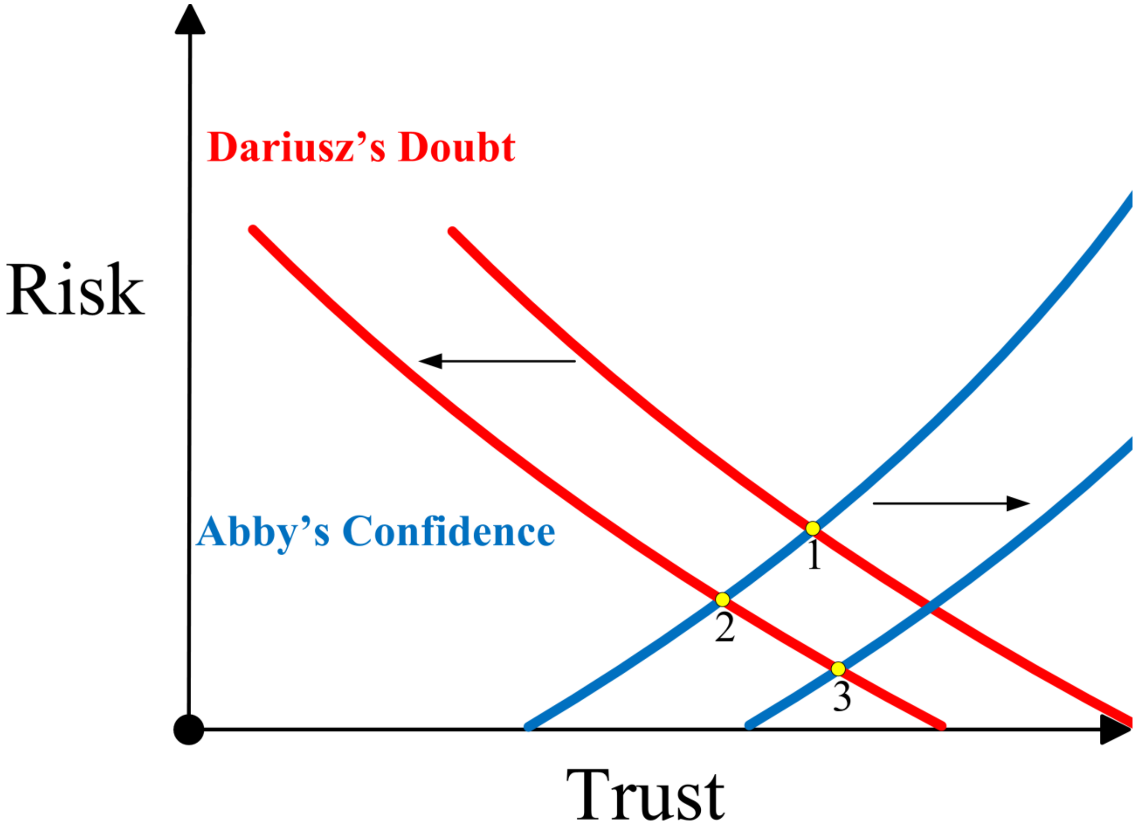


Figure 2.9. Abby Perceives Decrease in Dariusz's Doubt. Because Dariusz stops ignoring her bell rings, Abby interprets this as a decrease in Dariusz's doubt, causing the risk equilibrium to move from point 1 to point 2. Over time, Abby's confidence increases since she believes Dariusz is happy with her new behavior, resulting in the risk equilibrium move from point 2 to point 3.

CHAPTER THREE

COOPERATIVE TRUST GAMES

Synopsis

This chapter provides the theoretical development of the cooperative trust game: a new framework to study trust-based coalition formation in multi-agent systems using cooperative game theory as the underlying mathematical framework. We show how cooperative trust games can be used to study trust interaction outcomes between agents in coalitions as a result of their trust synergy and trust liability, and discuss how to apply these games for cooperative control in an autonomous military convoy.

This chapter begins with Section 3.1, which provides a high-level overview of trust in coalitions. Section 3.2 formally develops the theory of the cooperative trust game by stating its definition, characterizing different classes of trust games, and providing a means to study the division of the trust payoff among different members in a coalition. Section 3.3 provides a general trust game model that conforms to the theoretical constructions in Section 3.2. Section 3.4 applies the cooperative trust game model in Section 3.3 to the convoy application by defining the convoy trust game and proving the solution for the highest payoff coalition.

3.1 Background and Motivation

In Chapter Two, we saw how interactions between two agents can result in the formation of a trusting relationship, which can be leveraged for cooperative or collaborative activities. These types of relationships generally constrain individual-agent actions, since they imply that at least one contract (or mutual agreement) between the agents must exist. There is always some uncertainty as to whether or not either agent can or will satisfy some contract requirement – especially at the creation of a new contract. But in order to maintain the existence of a contract, each agent must overcome this uncertainty and assume that the other will do the same. The mechanism that facilitates this “act of faith” is regarded as trust. In essence, each agent in a relationship (whether a person or organization) mutually trusts that the loss of some control will result in cooperative gains that neither agent could achieve alone.

Since agents in an arbitrary multi-agent system are always assumed to have selfish interests, the goal of each agent is to try to find the most fruitful relationships in a pool of potential agents [112]. That said, we cannot assume that agents do not already have pre-existing relationships with other agents. Furthermore, some agents may actually be within strongly-connected sub-system groups known as **coalitions**, where every agent in each coalition has a relationship with every other agent in the same coalition. A coalition may contain a mixture of trustworthy and untrustworthy agents – but as a group, achieves cooperative gains that no sub-coalition could match. Thus, agents may be justified in forming relationships with coalition members who are not ideally trustworthy in order to acquire these cooperative gains as well.

As a simple example to illustrate this concept, consider two geographically-separated agents (who, perhaps, never physically met). Each agent would like to engage in a financial transaction in exchange for some good. One agent must provide the good (through the mail) and the other must provide the payment (through the mail or electronically). If both agents follow their economic best interest, then neither agent should participate in the transaction since both agents are vulnerable to betrayal by the other. This is because neither agent can truly verify the intent of the other agent before the other agent acts. Thus, if a transaction takes place, it can be entirely attributed to trust since both agents need to overcome the uncertainty associated with the transaction.

Let us suppose, however, that the value of the good and the size of the payment are sufficiently high such that no amount of mutual trust allows a direct transaction to take place. To handle this situation, both agents could form a coalition with a mutually trusted third party, such as an escrow agent. The escrow agent would receive the payment from one agent to verify that the good can be shipped, and then later disperse the payment to the other agent (minus the escrow fee) when the good has been verified as received. Here, each agent in the coalition benefits from the cooperative gains of the transaction. These gains would not be possible if even one agent chose to disband from the coalition.

This chapter intends to show how one could mathematically describe these types of trust-based interactions via the **cooperative trust game** to predict coalition formation and disbanding. It presents a rigorous treatment of coalition formation using cooperative game theory as the underlying mathematical framework. It is important to highlight that cooperative game theory is significantly different than the more widely recognized competitive (non-cooperative) game theory. Cooperative game theory focuses on what groups of self-interested agents can achieve. It is not concerned with how agents make choices or coordinate in coalitions, and does not assume that agents will always agree to follow arbitrary instructions. Rather, cooperative game theory defines games that tell how well a coalition can do for itself. And while the coalition is the basic modeling unit for a coalition game, the theory supports modeling individual agent preferences without concern for their possible actions. As such, it is an ideal framework for modeling trust-based coalition formation since it can show how each agent's trust preferences can influence a group's ability to reason about trustworthiness. We refer the reader to [126] for an excellent primer on cooperative game theory.

3.2 Theoretical Development

This section provides the theoretical development of the cooperative trust game. After formally defining the cooperative trust game in Section 3.2.1, we characterize different classes of trust games within the context of cooperative game theory. Our characterizations provide the necessary conditions for a cooperative trust game to be classified into a particular class. We discuss additive (Section 3.2.2) and constant-sum trust games (Section 3.2.3), which have limited value for cooperative applications, but are included for completeness. Afterward, we discuss superadditive (Section 3.2.4) and convex (Section 3.2.5) trust games, which show conditions for agents to form a grand coalition. In general, grand coalition solution concepts presented here can also be applied to smaller coalitions within a trust game through the use of a trust subgame.

Following the trust game classes, we provide some theoretical tools to analyze the division of the trust payoff between members in a trust-based coalition. We describe the notion of marginal contributions (Section 3.2.6) as well as the notions of altruistic and competitive contributions (Section 3.2.7) for convex trust games. We conclude by providing a means to define a cooperative trust game in multiple contexts using the multi-issue representation (Section 3.2.8).

3.2.1 Characteristic Payoff of the Cooperative Trust Game

Definition 3.1 (Cooperative Trust Game): Let $\Gamma = (N, v)$ be a cooperative trust game with transferable utility where:

- N is a finite set of agents, indexed by i
- $v: 2^N \rightarrow \mathbb{R}$ associates with each coalition $A \subseteq N$ a real-valued payoff $v(A)$ that is distributed between the agents. Singleton coalitions, by definition, are assigned no value; i.e. $v(i) = 0 \quad \forall i \in N$.

The transferable utility assumption means that payoffs in a coalition may be freely distributed among its members. With regards to payoff value of trust between agents, this assumption can be interpreted as a universal means for agents to mutually share the value of their trustworthy relationships. Trust cultivation often requires reciprocity between two agents as a necessary behavior to develop trust, and a transferable utility is a convenient way to model the exchange for this notion.

In defining a transferable payoff value of trust, one aspect to consider is the “goods of trust”. These refer to opportunities for cooperative activity, knowledge, and autonomy. In this chapter, we refer to these goods as **trust synergy** $s(A)$, which is a trust-based result that could not be obtained independently by two or more agents. We may also interpret trust synergy as the value obtained by agents in a coalition as a result of being able to work together due to their attitudes of trust for each other. In defining a set function for trust synergy, it is important to explicitly show how each agent’s attitude of trustworthiness for every other agent in a coalition affects this synergy. In general, higher levels of trust in a coalition should produce higher levels of synergy.

The payoff value of trust, however, also includes an opposing force in the form of vulnerability exposure, which we refer to as **trust liability** $l(A)$. Trusting involves being optimistic that the trustee will do something for the truster; and this optimism is what causes the vulnerability, since it restricts the truster's inferences about the likely actions of the trustee. However, the refusal to be vulnerable tends to undermine trust since it does not allow others to prove their own trustworthiness, stifling growth in trust synergy. Thus, we see that agents in trust-based relationships with other agents must be aware of the balance between the values of the trust synergy and trust liability in addition to their relative magnitudes.

Let the characteristic payoff function of a trust game be the difference between the trust synergy and the trust liability of a coalition A.

$$v(A) = s(A) - l(A) \tag{3.1}$$

This payoff is similar to the well-known constrained coalitional game (CCG) that incorporates gains from cooperation with the costs due to communications network restrictions [16]. However, the characteristic function v in CCGs is defined on the structure of a particular communications network between agents, whereas the characteristic function for our trust game is defined only on a set of agents. As such, agents who are completely disconnected from communication with other agents can still theoretically maintain membership in the same trust-based coalition.

3.2.2 Additive Trust Game

Additive games are considered inessential games in cooperative game theory since the value of the union of two disjoint coalitions ($A \cap B = \emptyset$) is equivalent to the sum of the values of each coalition.

$$v(A \cup B) = v(A) + v(B) \quad \forall A, B \subset N \tag{3.2}$$

We see that the total value of the trust relationships between any two disjoint coalitions must always be zero. In other words, the trust synergy between any two disjoint coalitions must always result in a value that is equal to their trust liability. Thus, by expanding Equation 3.2 for trust games and rearranging the terms, we can characterize an additive trust game as:

$$s(A \cup B) - l(A \cup B) = s(A) - l(A) + s(B) - l(B)$$

$$\{\forall A, B \subset N \colon A \cap B = \emptyset\}$$

$$s(A \cup B) - s(A) - s(B) = l(A \cup B) - l(A) - l(B) \tag{3.3}$$

$$\{\forall A, B \subset N \colon A \cap B = \emptyset\}$$

3.2.3 Constant-Sum Trust Game

In constant-sum games, the sum of all coalition values in N remains the same, regardless of any outcome.

$$v(N) = v(A) + v(N \backslash A) = \mathscr{k} \quad \forall A \subset N \tag{3.4}$$

Note that the notation $N \backslash A$ denotes a subset of N consisting of all coalition members except the members in A.

By expanding Equation 3.4 for trust games and rearranging the terms, we can see that the constant-sum trust game is a special case of a two-coalition additive trust game involving every agent in the game.

$$s(N) - l(N) = s(A) - l(A) + s(N\backslash A) - l(N\backslash A) \quad \forall A \subset N$$

$$s(N) - s(A) - s(N\backslash A) = l(N) - l(A) - l(N\backslash A) \quad \forall A \subset N \tag{3.5}$$

Definition 3.2 (Dummy Agent): An agent is a dummy agent if it's contribution to any coalition is exactly the amount that it is able to achieve alone.

Theorem 3.1: Γ is a constant-sum trust game implies that Γ is a zero-sum trust game.

Proof: If Γ is a constant-sum game, the following constraint for singleton coalitions must always hold:

$$s(N) - s(i) - s(N\backslash i) = l(N) - l(i) - l(N\backslash i) \quad \forall i \in N$$

By rearranging the terms, combining, and substituting, we get:

$$s(N) - l(N) = s(i) - l(i) + s(N\backslash i) - l(N\backslash i) \;\; \forall i \in N$$

$$v(N) = v(i) + v(N\backslash i) \quad \forall i \in N$$

$$v(N) = v(N\backslash i) \quad \forall i \in N$$

This implies that every agent in N must behave like a dummy agent if Γ is a constant-sum trust game. Since all agents behave like dummy agents and $v(i) = 0$ for all $i \in N$, then any coalition that forms in Γ will have no value. Hence, the value of the grand coalition is zero (i.e. $v(N) = \mathscr{k} = 0$). Therefore, the only possible constant-sum trust game is the zero-sum trust game. This completes the proof.

A trivial corollary from Theorem 3.1 worth noting is that Γ is a also a zero-sum trust game if $s(A) = l(A) \;\; \forall A \subset N$. This results in $v(A) = 0 \;\; \forall A \subset N$, thus making the grand coalition $v(N) = v(N\backslash A) = \mathscr{k} \;\; \forall A \subset N$. This result implies that every possible coalition in N must behave like a coalition of dummy agents in a constant-sum trust game and their combinations with other coalitions will yield no value. Hence, the value of the grand coalition is always zero (i.e. $v(N) = \mathscr{k} = 0$).

Theorem 3.1 shows that any constant-sum trust game is necessarily a zero-sum trust game, which is a special case of an additive trust game. These facts reinforce a notion that a group of agents who do not trust each other will always prefer to work as singleton coalitions. And even if there is some mutual trust between agents, gains from trust synergy are always lost to the trust liability, making it irrational to form any coalition with any other agent. Thus, if one determines that Γ is a constant-sum trust game, then this provides immediate justification for using non-cooperative game theory as the basis for modeling the purely competitive agents.

3.2.4 Superadditive Trust Game

In a superadditive game, the value of the union of two disjoint coalitions ($A \cap B = \emptyset$) is never less than the sum of the values of each coalition.

$$v(A \cup B) \geq v(A) + v(B) \quad \forall A, B \subset N \tag{3.6}$$

This implies a monotonic increase in the value of any coalition as the coalition gets larger.

$$A \subseteq C \subseteq N \rightarrow v(A) \leq v(C) \leq v(N) \tag{3.7}$$

This property of superadditivity tells us that the new links that are established between the agents in the two disjoint coalitions are the sources of the monotonic increases. This results in a snowball effect that causes all agents in the game to form the **grand coalition** (a coalition containing all agents in the game) since the total value of the new trust relationships between any two disjoint coalitions must always be positive semi-definite. In other words, the trust synergy between any two disjoint coalitions must always result in a value that is at least as large as their combined individual trust liabilities. Thus, by expanding the definition for trust games and rearranging the terms, we can characterize a superadditive trust game as:

$$s(A \cup B) - l(A \cup B) \geq s(A) - l(A) + s(B) - l(B)$$

$$\{\forall A, B \subset N: A \cap B = \emptyset\}$$

$$s(A \cup B) - s(A) - s(B) \geq l(A \cup B) - l(A) - l(B) \tag{3.8}$$

$$\{\forall A, B \subset N: A \cap B = \emptyset\}$$

3.2.5 Convex Trust Game

A game is convex if it is supermodular, and this trivially implies superadditivity (when $A \cap B = \emptyset$). Thus, we see that convexity is a stronger condition than superadditivity since the restriction that two coalitions must be disjoint no longer applies.

$$v(A \cup B) + v(A \cap B) \geq v(A) + v(B) \quad \forall A, B \subset N \tag{3.9}$$

In convex games, the incentive of joining a coalition grows as the coalition gets larger. This means that the marginal contribution of each agent $i \in N$ is non-decreasing.

$$v(A \cup i) - v(A) \leq v(C \cup i) - v(C) \text{ whenever } A \subset C \subset N \backslash i \tag{3.10}$$

Definition 3.3 (M-marginal Game): Given a game $\Gamma = (N, v)$ and a coalition $M \subseteq N$, the *M*-marginal game $v_M: 2^{N \backslash M} \to \mathbb{R}$ is defined by $v_M(A) = v(M \cup A) - v(M)$ for each $A \subseteq N \backslash M$.

Using Definition 3.3, Branzei, Dimitrov, and Tijs proved that a game is convex if and only if all of its marginal games are superadditive [19]. We provide their proof here as a means for the reader to readily justify this assertion.

Theorem 3.2: A game $\Gamma = (N, v)$ is convex if and only if for each $M \in 2^N$ the M-marginal game $(N \backslash M, v_M)$ is superadditive.

Proof:

Suppose (N, v) is convex. Let $M \subseteq N$ and $A, B \subseteq N \backslash M$. Then:

$$\begin{aligned} v_M(A \cup B) + v_M(A \cap B) &= v(M \cup A \cup B) + v(M \cup (A \cap B)) - 2v(M) \\ &= v\big((M \cup A) \cup (M \cup B)\big) + v\big((M \cup A) \cap (M \cup B)\big) - 2v(M) \\ &\geq v(M \cup A) + v(M \cup B) - 2v(M) \\ &= \big(v(M \cup A) - v(M)\big) + \big(v(M \cup B) - v(M)\big) \\ &= v_M(A) + v_M(B) \end{aligned}$$

where the inequality follows from the convexity of v. Hence, v_M is convex (and superadditive as well).

Now, let $A, B \subseteq N$ and $M = A \cap B$. Suppose that for each $M \in 2^N$, the game $(N \backslash M, v_M)$ is superadditive. If $M = \emptyset$, then the game $(N \backslash \emptyset, v_\emptyset) = (N, v)$ and $v(\emptyset) = 0$; hence, Γ is superadditive. If $M \neq \emptyset$, then because $(N \backslash M, v_M)$ is superadditive:

$$v_M\big((A \cup B) \backslash M\big) \geq v_M(A \backslash M) + v_M(B \backslash M)$$

$$v(A \cup B) - v(M) \geq v(A) - v(M) + v(B) - v(M)$$

$$v(A \cup B) + v(M) \geq v(A) + v(B)$$

$$v(A \cup B) + v(A \cap B) \geq v(A) + v(B)$$

This completes the proof.

By using the characterization in Theorem 3.2 and expanding it to our definition of a trust game, we can state a necessary requirement to produce a convex trust game: that the marginal trust synergy between any two coalitions must always result in a value that is at least as large as their marginal trust liability.

$$
\begin{gathered}
s_M((A \cup B)\backslash M) - l_M((A \cup B)\backslash M) \\
\geq s_M(A\backslash M) - l_M(A\backslash M) + s_M(B\backslash M) - l_M(B\backslash M) \\
\{\forall A, B \subset N \colon A \cap B = M\} \\
s_M((A \cup B)\backslash M) - s_M(A\backslash M) - s_M(A\backslash M) \\
\geq l_M((A \cup B)\backslash M) - l_M(A\backslash M) - l_M(B\backslash M) \\
\{\forall A, B \subset N \colon A \cap B = M\}
\end{gathered}
\tag{3.11}
$$

Convex games, in general, are convenient due to several nice, well-known properties [126].

- The core of a convex game is never empty.
- Convex games are totally balanced, meaning that their subgames are also convex, each with a non-empty core.
- Convex games have a stable set that coincide with its core.
- The Shapley value of a convex game is the barycenter of the core.
- The vertices of a core can be found in polynomial time using a polyhedron greedy algorithm [83].

3.2.6 Marginal Contribution in a Trust Game

An important question cooperative game theory can be used to answer is: how is the overall value of a coalition divided up among the different coalition members?

With our transferable utility assumption, we can intuitively think that the division of the coalition value is determined by the bargaining or sharing that occurs between the members. However, formally, we can analyze this division with the concept of marginal contribution. We use the concept of a subset team game to define marginal contribution.

Definition 3.4 (Subset Team Game): Given a game $\Gamma = (N, u)$ and a non-empty coalition $R \subseteq S \subseteq N$, the subset team game $u_R: 2^R \rightarrow \mathbb{R}$ associates a valued payoff $u_R(S)$ perceived by the agents in R when the agents in S cooperate.

Definition 3.5 (Marginal Contribution): Given a payoff function $u_R(S)$ in a subset team game, the marginal contribution of $R \subseteq S$ to a team S is $m_R(S) = u_S(S) - u_{S\backslash R}(S\backslash R)$.

With these definitions in place, we must now provide a way to connect these concepts to the cooperative trust game. To accomplish this, we must define a new notion called the subset trust game.

Definition 3.6 (Subset Trust Game): Given a cooperative trust game $\Gamma = (N, v)$ and a non-empty coalition $R \subseteq S \subseteq N$, the subset trust game $u_R: 2^R \rightarrow \mathbb{R}$ associates a trust payoff value $u_R(S)$ perceived by the agents in R when the agents in S cooperate:

$$u_R(S) = v(R) + \sum_{i \in R, j \in S\backslash R} v(\{i, j\}) \quad R \subseteq S \subseteq N \tag{3.12}$$

The rationale behind this payoff function is that the payoff has to be from the perspective of the agents in R. The agents in R can factor in the values related to the relationships between themselves (first term) and the relationships between agents in R and agents in S (second term). But they cannot factor in values related to the relationships between the agents in $S\backslash R$, since agents in R are assumed to have no direct knowledge of what is happening between the $S\backslash R$ agents.

3.2.7 Altruistic and Competitive Contribution Decomposition

In the analysis of a trust-based coalition, it may sometimes be useful to decompose the marginal contribution of a coalition subset even more finely. One way to do this is to leverage a framework developed by Arney and Peterson, where measures of cooperation are defined in terms of altruistic and competitive cooperation [8]. The unifying concept in this framework is a subset team game, which was defined in Definition 3.4.

Arney and Peterson limit the application of the framework to games where more agents in a coalition lead to more successful outcomes. Thus, adding more agents to a coalition should never reduce the coalition's payoff value. Also, the payoff value perceived by a coalition should not be smaller than the payoff value perceived by a subset of the same coalition. We refer to these two properties as fully-cooperative and cohesive, respectively.

Definition 3.7 (Fully-cooperative Property): A subset team game is fully-cooperative if $u_A(B) \leq u_A(C)$ for all $A \subseteq B \subseteq C \subseteq N$.

Definition 3.8(Cohesive Property): A subset team game is cohesive if $u_A(C) \leq u_B(C)$ for all $A \subseteq B \subseteq C \subseteq N$.

Now we define the meaning of competitive and altruistic contributions using the notation in this chapter.

Definition 3.9 (Altruistic Contribution): Given a payoff function $u_R(S)$ in a subset team game that is both cohesive and fully-cooperative, the altruistic contribution of $R \subseteq S \subseteq N$ is $a_R(S) = u_{S\backslash R}(S) - u_{S\backslash R}(S\backslash R)$.

Definition 3.10 (Competitive Contribution): Given a payoff function $u_R(S)$ in a subset team game that is both cohesive and fully-cooperative, the competitive contribution of $R \subseteq S \subseteq N$ is $c_R(S) = u_S(S) - u_{S\backslash R}(S)$.

Note that the marginal contribution decomposes as $m_R(S) = c_R(S) + a_R(S)$.

In order to use the new altruistic and competitive contribution definitions within a trust game, we must first show they relate to the cooperative game classes described earlier in this section. As it turns out, the fully-cooperative and cohesive properties conform to cooperative games that are convex.

Theorem 3.3: A subset team game that is both fully-cooperative and cohesive is a convex game.

Proof:

First, we prove the fully-cooperative case. If $u_A(B) \leq u_A(C)$ such that $A \subseteq B \subseteq C \subseteq N$, then the following inequalities are also true:

$$u_A(B) \leq u_A(B \cup i) \qquad A \subseteq B \subseteq N\backslash i$$

$$u_A(C) \leq u_A(C \cup i) \qquad A \subseteq C \subseteq N\backslash i$$

$$u_A(B \cup i) \leq u_A(C \cup i) \qquad A \subseteq B \subseteq C \subseteq N\backslash i$$

Since the system of inequalities shows that the contribution of an additional agent in a coalition is always non-decreasing, it is trivially true that:

$$u_A(B \cup i) - u_A(B) \leq u_A(C \cup i) - u_A(C) \quad A \subseteq B \subseteq C \subseteq N\backslash i$$

Next, we prove the cohesive case. If $u_A(C) \leq u_B(C)$ such that $A \subseteq B \subseteq C \subseteq N$, then the following inequalities are also true:

$$u_A(C) \leq u_{A\cup i}(C) \qquad A \subseteq C \subseteq N\backslash i$$

$$u_B(C) \leq u_{B\cup i}(C) \qquad B \subseteq C \subseteq N\backslash i$$

$$u_{A\cup i}(C) \leq u_{B\cup i}(C) \qquad A \subseteq B \subseteq C \subseteq N\backslash i$$

Since the system of inequalities shows that the contribution of an additional agent in a coalition subset is always non-decreasing, it is trivially true that:

$$u_{A\cup i}(C) - u_A(C) \leq u_{B\cup i}(C) - u_B(C) \quad A \subseteq B \subseteq C \subseteq N\backslash i$$

This completes the proof.

It is important to note that the additional agent i for both cases is never already inside either coalition B or C. If it was, then the proof would be invalid, as one could easily demonstrate counter examples under cases where an agent $i \in C\backslash B$.

Now that we have shown that a convex subset team game is fully-cooperative and cohesive, we may decompose the marginal contribution of a set of agents into both altruistic and competitive contributions whenever a trust game is convex. Using the payoff function $u_R(S)$ from Definition 3.6, we can calculate the $R \subseteq S \subseteq N$ altruistic contribution $a_R(S)$ and the competitive contribution $c_R(S)$ in coalition S.

$$a_R(S) = \sum_{i\in R, j\in S\backslash R} v(\{i,j\}) \qquad R \subseteq S \subseteq N \tag{3.13}$$

$$c_R(S) = v(S) - v(S\backslash R) - \sum_{i\in R, j\in S\backslash R} v(\{i,j\}) \qquad R \subseteq S \subseteq N \tag{3.14}$$

$$m_R(S) = a_R(S) + c_R(S) = v(S) - v(S\backslash R) \qquad R \subseteq S \subseteq N \tag{3.15}$$

3.2.8 Incorporating Multiple Contexts into a Trust Game

In practice, trust is often defined relative to some context. Context allows individuals to simplify complex decision-making scenarios by focusing on more narrow perspectives of situations or others, avoiding the potential for inconvenient paradoxes.

Cooperative trust games can also be defined relative to different contexts using the multi-issue representation [34], where we use the words “context” and “issue” interchangeably.

Definition 3.11 (Multi-issue Representation): A multi-issue representation is composed of a collection of cooperative games, each known as an issue, $\left(N^{(1)}, v^{(1)}\right), \left(N^{(2)}, v^{(2)}\right), \cdots, \left(N^{(\mathcal{k})}, v^{(\mathcal{k})}\right)$, which together constitute the cooperative game (N, v) where

- $N = N^{(1)} \cup N^{(2)} \cup \cdots \cup N^{(\mathcal{k})}$
- For each coalition $A \subseteq N$, $v(A) = \sum_{i=1}^{\mathcal{k}} v^{(i)}(A \cap N^{(i)})$

This approach allows us to define an arbitrarily complex trust game that can be easily decomposed into simpler trust games relative to a particular context. A set of agents in one context can overlap partially or completely with another set of agents in another context. And one can choose to treat the coalitional game in one big context, or the union of any number of contexts based on some decision criteria.

3.3 General Trust Game Model

In the previous section, we developed the theory for cooperative trust games without explicitly defining a trust game model. In this section, we provide a general model for trust games that conforms to the theoretical constructions in Section 3.2 and is adaptable to a wide variety of applications.

3.3.1 Managing Agent Trust Preferences

The attitude of trustworthiness that agents have toward other agents in a trust game is managed in an $|N| \times |N|$ matrix $\boldsymbol{T}$.

$$\boldsymbol{T} = \left[\boldsymbol{T}_{ij}\right]_{|N|\times|N|} = \begin{cases} \boldsymbol{T}_{ij} = 1, & i = j \\ \boldsymbol{T}_{ij} \in [0,1], & i \neq j \end{cases} \qquad (3.16)$$

This matrix is populated with values $\boldsymbol{T}_{ij}$ that represent the probability that agent j is trustworthy from the perspective of agent i. The values $\boldsymbol{T}_{ij}$ can also be interpreted as the probabilities that agent i will allow agent j to interact with it, since rational agents would prefer to interact with more trustworthy agents.

The manner in which $\boldsymbol{T}_{ij}$ is evaluated depends on an underlying trust model. We make no assumption about the use of a particular trust model (other than it being probabilistic), as the choice of an appropriate model may be application-specific. We also make no assumption about the spatial distribution of the agents in a game. However, we do assume that each agent i does fully trust itself at all times, and express this notion through the “ones” in the diagonal.

3.3.2 Modeling Trust Synergy and Trust Liability

We now provide a general model for trust synergy and trust liability that can be adapted for a variety of applications. Our model makes use of a symmetric matrix $\boldsymbol{\Sigma}$ to manage potential trust synergy and a matrix $\boldsymbol{\Lambda}$ to manage potential trust liability. $\boldsymbol{\Sigma}$ is symmetric because we assume that agents mutually agree on the benefits of a synergetic interaction.

$$\boldsymbol{\Sigma} = \left[\boldsymbol{\Sigma}_{ij}\right]_{|N|\times|N|} = \begin{cases} \boldsymbol{\Sigma}_{ij} = 0, & i = j \\ \boldsymbol{\Sigma}_{ij} = \boldsymbol{\Sigma}_{ji} \geq 0, & i \neq j \end{cases} \tag{3.17}$$

$$\boldsymbol{\Lambda} = \left[\boldsymbol{\Lambda}_{ij}\right]_{|N|\times|N|} = \begin{cases} \boldsymbol{\Lambda}_{ij} = 0, & i = j \\ \boldsymbol{\Lambda}_{ij} \geq 0, & i \neq j \end{cases} \tag{3.18}$$

As with the $\boldsymbol{T}$ matrix, we make no assumptions about how $\boldsymbol{\Sigma}$ and $\boldsymbol{\Lambda}$ are calculated, since the meaning of their values may depend on the application. For example, the calculations for $\boldsymbol{\Sigma}_{ij}$ and $\boldsymbol{\Lambda}_{ij}$ between two agents may not only take into account each agent's individual intrinsic attributes – it may also factor in externalities (i.e. political climate, weather conditions, pre-existing conditions, etc.) that neither agent has direct control over.

Definition 3.12 (Trust Synergy Model): The total value of the trust synergy in a coalition is defined as the following set function:

$$s(A) = \sum_{i,j\in A} \boldsymbol{\Sigma}_{ij}\, \boldsymbol{T}_{ij}\boldsymbol{T}_{ji} \quad \forall i > j \tag{3.19}$$

Trust synergy is the value obtained by agents in a coalition as a result of being able to work together due to their attitudes of trust for each other. The set function $s(A)$ assumes that the actions "agent i allows agent j to interact" and "agent j allows agent i to interact" are independent. This is reasonable since agents are assumed to behave as independent entities within a trust game (i.e. no agent is controlled by any other agent). Therefore, we treat the product $\boldsymbol{T}_{ij}\boldsymbol{T}_{ji}$ as the relative strength of a trust-based synergetic interaction (not a probability), which justifies the use of the summation. The value for $\boldsymbol{\Sigma}_{ij}$ serves as a weight for a trust-based synergetic interaction.

Definition 3.13 (Trust Liability Model): The total value of the trust liability in a coalition is defined as the following set function:

$$l(A) = \sum_{i,j \in A} \boldsymbol{\Lambda}_{ij}\boldsymbol{T}_{ij} \qquad \forall i \neq j \tag{3.20}$$

Trust liability can be thought of as the vulnerability that agents in a coalition expose themselves to due to their attitudes of trust for each other. We treat the product $\boldsymbol{\Lambda}_{ij}\boldsymbol{T}_{ij}$ as a measure for agent i's exposure to unfavorable trust-based interactions from agent j. A high amount of trust can expose agents to high levels of vulnerability. But each agent can regulate its exposure to trust liability by adjusting $\boldsymbol{T}_{ij}$. Changes to $\boldsymbol{T}_{ij}$, however, also influence the benefits of trust synergy.

With the trust synergy and trust liability defined, we can now define the trust payoff function in Equation 3.1 as the difference between the trust synergy and trust liability.

$$v(A) = \sum_{\substack{i,j \in A \\ \forall i > j}} \boldsymbol{\Sigma}_{ij}\, \boldsymbol{T}_{ij}\boldsymbol{T}_{ji} - \sum_{\substack{i,j \in A \\ \forall i \neq j}} \boldsymbol{\Lambda}_{ij}\boldsymbol{T}_{ij} \tag{3.21}$$

$$v(A) = \sum_{\substack{i,j \in A \\ \forall i > j}} \boldsymbol{T}_{ij}\boldsymbol{T}_{ji}\left(\boldsymbol{\Sigma}_{ij} - \frac{\boldsymbol{\Lambda}_{ij}}{\boldsymbol{T}_{ji}} - \frac{\boldsymbol{\Lambda}_{ji}}{\boldsymbol{T}_{ij}}\right) \tag{3.22}$$

The factorization in (3.22) shows us that the first factor ($\boldsymbol{T}_{ij}\boldsymbol{T}_{ji}$) will always be greater than or equal to zero while the second factor can be either positive or negative. Hence, by isolating the second factor and recognizing that trust values equal to 1 produce the smallest possible reduction in the second factor, we can state the condition that guarantees the potential for two agents to form a trust-based pair coalition.

Proposition 3.1: Any two agents $i, j \in N$ will never form a trust-based pair coalition if $\boldsymbol{\Sigma}_{ij} < \boldsymbol{\Lambda}_{ij} + \boldsymbol{\Lambda}_{ji}$. Otherwise, the potential exists for agent i and j to form a trust-based pair coalition.

Proposition 3.2: If two agents can never form a trust-based pair coalition, then the best strategy for both agents is to never trust each other (i.e. $\boldsymbol{T}_{ij} = \boldsymbol{T}_{ji} = 0$).

It is important to note that Proposition 3.1 does not extend to trust-based coalitions larger than two due to the non-trivial coupling of trust dynamics between different agents as coalitions grow larger. For example, two agents who may produce a negative trust payoff value as a pair may actually realize a positive trust payoff with the addition of a third agent. This situation occurs if both agents have positive trust relationships with the third agent that outweighs their own negative trust relationship. Such a situation is common in real world scenarios, and justifies the importance and value of trusted third parties, such as escrow companies, website authentication services, and couples therapists.

3.4 Convoy Trust Game

In this section, we present an example of a cooperative trust game for a specific application: the autonomous convoy. We define the convoy trust game, which describes a cooperative trust game where the agents intend to move forward together in a single file. This type of game can be naturally adapted to the analysis of traffic patterns, general leader-follower applications, hierarchical organizations, or applications with sequential dependencies. Our overall goal in this section is to understand how trust-based coalitions will form under this type of scenario.

3.4.1 4-Agent Convoy Trust Game

We begin with a simple convoy scenario that models a four-agent convoy, $N = \{1,2,3,4\}$, which intends to move together in a single file. The value of each index in N also represents the agent's position in the convoy. For this scenario, we interpret the trust synergy in the coalition to represent the agents in the coalition moving forward. Thus, we set the values in the trust synergy matrix **Σ** equal to the number of agents that will move forward if the two agents are moving forward (inclusive of the two agents). We interpret the trust liability in the coalition to represent the vulnerability of agents in the coalition to stop moving. Thus, we set the values in the trust liability matrix **Λ** equal to the number of agents that can prevent a particular agent from moving forward in an agent coalition pair.

The values in **Σ** and **Λ** for a 4-agent convoy trust game are:

$$\boldsymbol{\Sigma} = \begin{bmatrix} 0 & 2 & 3 & 4 \\ 2 & 0 & 3 & 4 \\ 3 & 3 & 0 & 4 \\ 4 & 4 & 4 & 0 \end{bmatrix} \quad \boldsymbol{\Lambda} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 \\ 2 & 2 & 0 & 2 \\ 3 & 3 & 3 & 0 \end{bmatrix} \tag{3.23}$$

It is important to note that this convoy trust game only considers one context for coalition formation. Additional contexts (such as the presence of hostile forces, the time of day, and weather conditions) could also influence the overall trust between agents. If applicable, these contexts would need to be modeled separately and possibly combined using the multi-issue representation described in Section 3.2.8.

3.4.2 Analysis of the 4-Agent Convoy Trust Game

First, let us analyze the 4-agent convoy trust game as an additive trust game. While there are infinitely many solutions for $\boldsymbol{T}$ that conform to Equation 3.4, the most obvious solution is the extreme situation where no vehicle trusts any other vehicle – or, when $\boldsymbol{T}$ is the identity matrix ($\boldsymbol{T} = \boldsymbol{I}$). In this case, it can clearly be seen from Equation 3.21 that no vehicle will ever affect another vehicle, either positively or negatively. Thus, each vehicle will ultimately form a singleton coalition and fail to work cooperatively with any other vehicle.

Next, let us analyze another extreme situation where every vehicle completely trusts every other vehicle – or, when $\boldsymbol{T} = [1]_{4\times4}$. As such, we can enumerate the trust payoff values for each possible coalition.

$$v(\{1,2\}) = 1; \quad v(\{1,3\}) = 1; \quad v(\{1,4\}) = 1; \quad v(\{2,3\}) = 0;$$

$$v(\{2,4\}) = 0; \quad v(\{3,4\}) = -1; \quad v(\{1,2,3\}) = 2;$$

$$v(\{1,2,4\}) = 2; \quad v(\{1,3,4\}) = 1; \quad v(\{2,3,4\}) = -1;$$

$$v(\{1,2,3,4\}) = 2;$$

These results provide us an interesting insight, in that all vehicles behind the lead vehicle find higher values of trust payoff with the lead vehicle than with the nearest vehicle. As such, as long as the lead vehicle is a member of a coalition in this game, there will be no incentive for any other vehicle to abandon the coalition. Thus, the vehicles ultimately form the grand coalition. Note, however, that the formation of a grand coalition does not imply that the trust game is superadditive or convex. This assertion is justified with the observation that $v(\{3,4\}) \ngeq v(\{3\}) + v(\{4\}) = 0$.

In order to form a convex 4-agent convoy trust game, we must satisfy the condition in Equation 3.11, which ensures that all trust payoff values in any coalition are at least as large as any sub-coalition. While there are infinitely many solutions for $\boldsymbol{T}$ that conform to Equation 3.11, the games with the highest trust payoff have either one of the following trust matrices (proven in the next section)

$$\boldsymbol{T}^{(1)} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \end{bmatrix} \quad \boldsymbol{T}^{(2)} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 \end{bmatrix}$$

$$\boldsymbol{T}^{(3)} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \end{bmatrix} \quad \boldsymbol{T}^{(4)} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 \end{bmatrix}$$

$\boldsymbol{T}^{(1)}$, $\boldsymbol{T}^{(2)}$, $\boldsymbol{T}^{(3)}$, and $\boldsymbol{T}^{(4)}$ are modified versions of $[1]_{4\times 4}$ and all produce the same results in the trust payoff value function. The main modification, following from Proposition 3.1, ensures that vehicles 3 and 4 have no trust toward each other since the trust liabilities between them always outweigh their trust synergies. The following is the enumeration of the trust payoff values for the 4-agent convoy trust game with any of these trust matrices:

$$v(\{1,2\}) = 1; \quad v(\{1,3\}) = 1; \quad v(\{1,4\}) = 1; \quad v(\{2,3\}) = 0;$$

$$v(\{2,4\}) = 0; \quad v(\{3,4\}) = 0; \quad v(\{1,2,3\}) = 2; \quad v(\{1,2,4\}) = 2;$$

$$v(\{1,3,4\}) = 2; \quad v(\{2,3,4\}) = 0; \quad v(\{1,2,3,4\}) = 3;$$

Because we now have a convex trust game, we have the ability to analyze these results in terms of altruistic and competitive contributions. In other words, we can get better insight into the core components that make up each subset team's marginal contribution to a coalition. For example, if we wished to understand the contributions of coalition $\{1,2\}$ to the coalition $\{1,2,3\}$, then we can calculate the altruistic contribution by $a_{\{1,2\}}(\{1,2,3\}) = v(1,3) + v(2,3) = 1$ and the competitive contribution by $c_{\{1,2\}}(\{1,2,3\}) = v(1,2,3) - v(3) - a_{\{1,2\}}(\{1,2,3\}) = 1$. Thus, we can clearly see that half of the marginal contribution of coalition $\{1,2\}$ to coalition $\{1,2,3\}$ is an altruistic contribution with agent $\{3\}$.

The deep insight we gain from analyzing the highest-payoff results is that "all vehicles behind the lead vehicle need only trust the lead vehicle in the convoy to move forward, provided the lead vehicle trusts every other vehicle to follow it." This echoes the intuition seen in Jean-Jacques Rousseau's classic "stag hunt" game, where there is no incentive for any player to cheat by not cooperating as long as each player can trust others to do the same [40].

For autonomous convoys, our results suggest that follower vehicles only need to communicate with the lead vehicle to ensure trustworthy coalition stability when cooperating to move forward. This hub-and-spoke communications network would, therefore, foster the reciprocity that is necessary to cultivate trust between the leader and its followers while also keeping the computational complexity of the network to a minimum of $O(n)$. In effect, the lead vehicle serves as the trusted third-party for all of the follower vehicles, just like the escrow agent served as the trusted third-party for the buyer and seller in our example at the beginning of this chapter. The presence of the solution $\boldsymbol{T}^{(4)}$ suggests that trust-based redundancy can also be achieved with the second vehicle in the event of a catastrophic failure to the lead vehicle. The cost of the trust-based redundancy would require an additional $(|N| - 2)$ point-to-point connections, but the computational complexity would not change.

3.4.3 Highest Payoff Solution to the N-Agent Convoy Trust Game

We conclude this section by generalizing the convoy trust game for any number of vehicles and prove the solution for the highest payoff trust-based coalition.

Definition 3.14 (N-agent Convoy Trust Game): The values in **Σ** and **Λ** for a convoy trust game with $|N|$ agents are:

$$\mathbf{\Sigma} = \left[\mathbf{\Sigma}_{ij}\right]_{|N|\times|N|} = \begin{cases} \mathbf{\Sigma}_{ij} = 0, & i = j \\ \mathbf{\Sigma}_{ij} = \max(\{i, j\}), & i \neq j \end{cases} \tag{3.24}$$

$$\mathbf{\Lambda} = \left[\mathbf{\Lambda}_{ij}\right]_{|N|\times|N|} = \begin{cases} \mathbf{\Lambda}_{ij} = 0, & i = j \\ \mathbf{\Lambda}_{ij} = i - 1, & i \neq j \end{cases} \tag{3.25}$$

Theorem 3.4: The N-agent convoy trust game that produces the grand coalition with the highest payoff value has a trust matrix that conforms to the following construction:

$$\boldsymbol{T} = \left[\boldsymbol{T}_{ij}\right]_{|N|\times|N|} = \begin{cases} \boldsymbol{T}_{ij} = 1, & i = j \\ \boldsymbol{T}_{ij} = 1, & i \neq j, \min(\{i, j\}) = 1 \\ \boldsymbol{T}_{ij} = \boldsymbol{T}_{ji} \in \{0,1\}, & i \neq j, \min(\{i, j\}) = 2 \\ \boldsymbol{T}_{ij} = 0, & i \neq j, \min(\{i, j\}) > 2 \end{cases} \tag{3.26}$$

Proof:

Suppose we generalize the values in **Σ** and **Λ** according to Equations 3.24 and 3.25, respectively. According to Proposition 3.1, two agents $i, j \in N$ will never form a trust-based coalition pair if $\mathbf{\Sigma}_{ij} < \mathbf{\Lambda}_{ij} + \mathbf{\Lambda}_{ji}$. Thus, by substitution:

$$\max(\{i, j\}) < (i - 1) + (j - 1)$$

$$\max(\{i, j\}) < i + j - 2$$

We see that if i is the maximum value, then $0 < j - 2$. Similarly, if j is the maximum value, then $0 < i - 2$. Thus, the inequalities tell us that any vehicle behind the second vehicle will never form a trust-based coalition with any other vehicle behind the second vehicle. Therefore, by Proposition 3.2, the best strategy for these vehicles is to have no trust for each other; hence $\boldsymbol{T}_{ij} = 0$ when $\min(\{i, j\}) > 2$ for $i \neq j$.

The inequalities above also imply that trust-based coalition formation is possible with either the lead agent or the second agent. Using Equation 3.22 and our definitions in Equations 3.24 and 3.25, the trust payoff values for a coalition in the convoy trust game are:

$$v(A) = \sum_{\substack{i,j \in A \\ \forall i > j}} \boldsymbol{T}_{ij}\boldsymbol{T}_{ji}\left(\max(\{i, j\}) - \frac{i-1}{\boldsymbol{T}_{ji}} - \frac{j-1}{\boldsymbol{T}_{ij}}\right)$$

We may now define trust payoff values for any pair of agents as:

$$v(\{i, j\}) = \boldsymbol{T}_{ij}\boldsymbol{T}_{ji}\left(\max(\{i, j\}) - \frac{i-1}{\boldsymbol{T}_{ji}} - \frac{j-1}{\boldsymbol{T}_{ij}}\right)$$

Let us first analyze the coalition formation with the lead agent. If $i = 1$, then $\max(\{i, j\}) = j$. Therefore, the payoff value for a pair coalition between i and j is:

$$v(\{1, j\}) = \boldsymbol{T}_{1j}\boldsymbol{T}_{j1}\left(j - \frac{j-1}{\boldsymbol{T}_{1j}}\right)$$

$$v(\{1, j\}) = j\boldsymbol{T}_{1j}\boldsymbol{T}_{j1} - j\boldsymbol{T}_{j1} + \boldsymbol{T}_{j1}$$

$$v(\{1, j\}) = \boldsymbol{T}_{j1}\left(j\boldsymbol{T}_{1j} - j + 1\right)$$

By inspection, we see that the highest trust payoff value is achieved when both the lead agent and any other agent fully trust each other (i.e., when $\boldsymbol{T}_{1j} = \boldsymbol{T}_{j1} = 1$).

However, to justify this assertion, we must also show that this assertion is true when $j = 1$. If $j = 1$, then $\max(\{i, j\}) = i$. Therefore, the payoff value for a pair coalition between i and j is:

$$v(\{i,1\}) = \boldsymbol{T}_{i1}\boldsymbol{T}_{1i}\left(i - \frac{i-1}{\boldsymbol{T}_{1i}}\right)$$

$$v(\{i,1\}) = i\boldsymbol{T}_{i1}\boldsymbol{T}_{1i} - i\boldsymbol{T}_{i1} + \boldsymbol{T}_{i1}$$

$$v(\{i,1\}) = \boldsymbol{T}_{i1}(i\boldsymbol{T}_{1i} - i + 1)$$

Again, by inspection, we confirm that the highest trust payoff is achieved when both the lead agent and any other agent completely trust each other. Therefore, $\boldsymbol{T}_{ij} = 1$ when the $\min(\{i, j\}) = 1$ for $i \neq j$.

Now, we analyze coalition formation with the second vehicle. If $i = 2$, then $\max(\{i, j\}) = j$. Therefore, the payoff value for a pair coalition between i and j is:

$$v(\{2,j\}) = \boldsymbol{T}_{2j}\boldsymbol{T}_{j2}\left(j - \frac{1}{\boldsymbol{T}_{j2}} - \frac{j-1}{\boldsymbol{T}_{2j}}\right)$$

$$v(\{2,j\}) = \boldsymbol{T}_{2j}\boldsymbol{T}_{j2}j - \boldsymbol{T}_{2j} - j\boldsymbol{T}_{j2} + \boldsymbol{T}_{j2}$$

$$v(\{2,j\}) = \boldsymbol{T}_{j2}\big(j\boldsymbol{T}_{2j} - j + 1\big) - \boldsymbol{T}_{2j}$$

The highest trust payoff that can be achieved with the second vehicle is equal to zero, and this only occurs when both vehicles either have complete trust in each other (i.e., when $\boldsymbol{T}_{2j} = \boldsymbol{T}_{j2} = 1$) or no trust in each other (i.e., when $\boldsymbol{T}_{2j} = \boldsymbol{T}_{j2} = 0$). Any other combination of trust values will produce negative trust payoff values. However, to justify this assertion, we must also show that this assertion is true when $j = 2$. If $j = 2$, then $\max(\{i, j\}) = i$. Therefore, the payoff value for a pair coalition between i and j is:

$$v(\{i,2\}) = \boldsymbol{T}_{i2}\boldsymbol{T}_{2i}\left(i - \frac{i-1}{\boldsymbol{T}_{2i}} - \frac{1}{\boldsymbol{T}_{i2}}\right)$$

$$v(\{i,2\}) = i\boldsymbol{T}_{i2}\boldsymbol{T}_{2i} - i\boldsymbol{T}_{i2} + \boldsymbol{T}_{i2} - \boldsymbol{T}_{2i}$$

$$v(\{i,2\}) = \boldsymbol{T}_{i2}(i\boldsymbol{T}_{2i} - i + 1) - \boldsymbol{T}_{2i}$$

By inspection, we confirm that the highest trust payoff that can be achieved with the second vehicle is equal to zero. Therefore, $\boldsymbol{T}_{ij} = \boldsymbol{T}_{ji} \in \{0,1\}$ when $\min(\{i,j\}) = 2$ for $i \neq j$.

To complete the proof, we simply state our assumption that each vehicle fully trusts itself, since it is impossible for a vehicle to diverge from a singleton coalition. Therefore, $\boldsymbol{T}_{i,j} = 1$ when $i = j$. This completes the proof.

Conclusion

In summary, this chapter defined and developed the cooperative trust game, which formalizes the study of coalition formation with trust-based interactions using cooperative game theory. We characterized different classes of cooperative trust games, provided a general model for cooperative trust games, and showed how the model could be applied to an autonomous convoy application within the context of moving forward together. Our main result from the application proves that all vehicles behind the lead vehicle in a convoy need to only trust the lead vehicle (and no other vehicle) to move forward, so long as the lead vehicle trusts every other vehicle to follow it. In other words, the most optimal trust payoff occurs when the lead vehicle acts as the trusted third-party between all of the follower vehicles.

CHAPTER FOUR

COMPUTATIONAL TRUST IN MULTI-AGENT SYSTEMS

Synopsis

This chapter reviews relevant computational trust research in multi-agent systems, and summarizes results and conclusions from this research. It is not intended to be an exhaustive review of all computational trust research – rather, its purpose is to provide the reader with sufficient background about this research domain to understand the relevance of the contributions in later chapters.

The chapter begins with Section 4.1, which focuses its attention on mobile ad hoc networks (MANETs) to highlight different types of trust-based attacks. Section 4.2 provides high-level descriptions of computational trust definitions, metrics, and properties found in the literature. Section 4.3 discusses generally what computational trust models take into account and how they are implemented within a network. Sections 4.4, 4.5, and 4.6 present a brief history of direct trust models, recommendation trust models, and hybrid trust models, respectively. Section 4.7 focuses on general mechanisms that bring about various trust dynamics within a network. Section 4.8 discusses the concept of system-level trust and how protocols of interaction can be established to ensure agents lose utility if they do not follow the rules of the system.

4.1 Overview

In Chapter Three, our work with cooperative trust games provided a means to study the outcomes of trust-based interactions in coalitions. However, in studying these outcomes, we assumed that matrix $\boldsymbol{T}$ in Equation 3.16 was given and static. We never made any assumptions about how matrix $\boldsymbol{T}$ was formed nor introduced any forms of trust dynamics. Our only requirement was that the individual elements in matrix $\boldsymbol{T}$ lay within the bounds of a probabilistic value.

It turns out, however, that there is a rich body of research dedicated to computationally determining a value for trust. This research is motivated by the need for soft security [42] [54] – a requirement to defend against the threat of unwanted or undesired behavioral changes in a system. Often, soft security is intended to complement hard security methods, like cryptography protection, since hard security cannot protect against illegitimate behaviors after a hard security event (such as file decryption). Trust management is a subset of the soft security research area and helps agents to evaluate the trade-off between security and performance when dealing with other agents.

This chapter focuses on trust management in multi-agent systems. The literature discusses trust in domains such as wireless sensor networks [95] [96], social networks [22] [73], and internet applications [55] [64]. However, our discussion of computational trust highlights trust management in mobile ad hoc networks (MANETs) [13] [31] [79]. This particular multi-agent system lends itself well to military robotics applications and allows us to underscore the different types of security vulnerabilities that can be exploited by military's adversaries.

4.1.1 Challenges with MANETs

MANETs are groups of mobile agents which can self-configure and form wireless communication networks without the need of a fixed infrastructure or centralized control authority [100]. They are able to be deployed quickly without any advanced planning for expensive network infrastructure, making them ideal for military applications, emergency rescue operations, undersea operations, environmental monitoring, and space exploration. Unfortunately, within such a network, it is often difficult to ensure secure communications. Agents are susceptible to passive eavesdropping, active interference, data tampering, information leakages, impersonation, and message replay.

In addition to securing communications, MANETs face other difficulties in practice. Namely, agents often have considerable constraints in bandwidth, computing power, and energy [35]. In addition, agents are often deployed in harsh or uncontrolled environments, thereby increasing the likelihood of security compromises and agent malfunctions. Because of all of these challenges, it is essential for agents to have the ability to quantify trust in observed behaviors of other agents to ensure productive collaborative and cooperative activities.

4.1.2 Information-Based Attacks towards MANETs

Information-based attacks are dominantly considered in the literature for trust management schemes in MANETs. This is because the dominate context for trust in MANETs relates to reliable bi-directional communication between nodes. Agents are generally modeled as mobile nodes with the sole ability to send and receive information packets wirelessly. Additional sensing capabilities, such as vision or localization, are often implied as necessary for both mobility and information-gathering, but rarely used directly in a trust scheme itself. Specific applications for MANET trust schemes include secure routing [51] [121], authentication [103], intrusion detection [5] [6] [87], access control [3], and key management [29].

The literature classifies information-based attacks a number of ways. Liu et al describe a classification based on passive and active attacks, which characterize attacks by both the nature of the attack and the type of attacker [81]. Passive attacks occur when unauthorized agents gain access to an asset in the MANET, but do not modify any content or behavior in the asset. Examples of passive attacks include eavesdropping and traffic flow analysis. Active attacks, on the other hand, occur when unauthorized agents intentionally influence the network in a nefarious manner. This may take the form of modifying or replaying messages, impersonating another agent, or consuming an excess amount of resources in the network.

Attacks can also be categorized by the legitimacy of the agent in the network, which Wu et al described as insider and outsider attacks [149]. An insider attack is done by an agent who is authorized to access a network, but uses the network resource in a malicious way. Insiders generally attempt to exploit bugs or poorly configured privileges. Outsider attacks, on the other hand, are initiated by an unauthorized agent who intends to carry out insider attacks through a stolen authorized account.

Levien categorizes attacks in a more general fashion based on the graph of a trust network [78]. Attacks are considered either as edge attacks or node attacks. Edge attacks are constrained in the sense that only one false opinion can be generated for each edge attack. This type of attack can be thought of as creating a false edge within the trust graph. Node attacks are more powerful however, and amount to a node being impersonated by a malicious node, resulting in the potential for many edge attacks.

There are numerous ways an attacker can disrupt the functionality of a MANET. We provide a representative, but non-exhaustive, list of trust-based attacks against MANETs. This list intends to show the diversity of potential attacks that trust schemes may need to defend against to ensure efficient and secure communications.

- **False Recommendation Attack (FRA)**. In a FRA, a malicious node provides false recommendations to isolate good nodes from the network. In a similar “stacking attack”, a malicious node keeps complaining about another node to establish a negative reputation for the other node. A trust scheme’s ability to aggregate multiple recommendations from multiple nodes can reduce the influence of such an attack [135].
- **On-Off Attack (OOA)**. In an OOA, a malicious node alternates between behaving well and badly, depending on the importance of the situation. Its goal is to stay undetected while disrupting services. Handling this attack can be done by weighting older observations less than newer observations, and aggregating many different observations from multiple sources into a trust scheme to reduce the influence of such an attack.
- **Conflicting Behavior Attack (CBA)**. In a CBA, a malicious node behaves differently to different groups of nodes with the intent to create a conflict between the groups. For example, a malicious node may provide a positive recommendation about a node to one group, but a negative recommendation

about the same node to a different group. This results in confusion and non-trusted relationships, which impacts the effectiveness of communications within a network. A CBA can be handled in much the same way as an OOA.

- **Camouflage Attack**. In a camouflage attack, a malicious node attempts to build up trust by behaving similarly to the observed majority of nodes. Then, after enough trust has been earned, it begins to behave badly for specific occasions. This attack is often difficult to detect, especially if the bad behaviors do not frequently occur or penalties from other nodes are relatively low. Generally, a centralized trust scheme has the best chance of noticing these types of attacks since it has access to all observations about every node in the network.
- **Collusion Attack**. In a collusion attack, multiple malicious nodes collaborate to give false recommendations about good nodes. In this sense, it is very similar to the FRA, but more difficult to defend against. Direct observations of the good node under attack often provide the best defense against collusion attacks; however, because of the mobile nature of MANETs, it may be difficult to maintain vigilance against motivated adversaries.
- **Newcomer / Sybil Attack**. Newcomer and Sybil attacks are similar in the sense that they try to make good nodes misidentify the malicious node, thereby making past trust measurements obsolete. For a newcomer attack, a malicious node attempts to discard its bad reputation by leaving a system

and later rejoining it as a ‘new user’, thereby flushing out its previous history. For a Sybil attack, a malicious node claims and controls multiple identities, and ruins the reputation of the stolen identities. This type of attack affects topology maintenance and fault tolerant schemes, such as multi-path routing. Trust schemes without a centralized administrative node are particularly vulnerable to both types of attacks.

4.2 Computational Trust Definitions, Metrics, and Properties

A universally-accepted definition of computational trust has not been established [54]. This may be due to the abstract nature of trust, but more likely, it is a reflection of the variety of computational models used to estimate trustworthiness. This being said, trust definitions can be broadly segmented into the following categories:

- **Definition based on probability**. Trust defined as a probability measure interprets trust to be the probability that another agent will perform some action within a specific time in a specific context [48][73] [74] [143].
- **Definition based on belief**. Trust defined as a belief interprets trust as the willingness to act on the basis of another’s actions or opinions [27] [90]. These beliefs are generally based on probabilities for related actions and opinions.
- **Definition based on transitivity**. Trust defined as a transitive relationship interprets trust as a weighted binary relation between two members in a network [146].

Trust metrics are used to evaluate and compare trust in different contexts. In every reviewed case, it is regarded as a relative factor that is represented as one of the following:

- **Scaled Value.** Represented as a continuous or discrete value within some range to measure the level of trust [114]. Lower values generally refer to low trust or explicit distrust; high values refer to high trust.
- **Multi-faceted representation.** Represented as a combination of values. For example, a trust metric can be represented as a combination of a trust value and a confidence measure [137]. Another metric represents trust as a triplet of belief, disbelief, and uncertainty [72].
- **Logical metric.** Represented as a value that is a result of some logical or application-specific calculation. Some approaches use probability as a metric [59] [111]. Others use ratios of good and bad results to estimate trust [155]. Fuzzy logic has also been used to associate labels from natural language to trust values [43].

The literature also describes certain properties of trust that are frequently found in trust networks [31] [54].

- **Dynamicity**. This property says that trust is based on changing temporal and spatial local information, and therefore, is never static.
- **Subjectivity**. This property implies that different trusters can determine different levels of trust against the same trustee due to different private biases, world views, and experiences.

- **Asymmetry**. This property says that trust is unidirectional between agents. So agent i can trust agent j to some level, but agent j does not necessarily need to trust agent i to the same level.
- **Transitivity**. This property implies that trust can be passed along a path of trusting nodes. So if agent i trusts agent j, and agent j trusts agent k, then agent i can trust agent k to a certain level. However, in order to use transitivity between two agents to a third party, a truster must maintain two types of trust: trust in the trustee and trust in the trustee's recommendation of the third party.
- **Composiblity**. This property means that trust information received from all available paths can be composed together to obtain a single trust value.
- **Context-Dependency**. This property provides the meaning behind a trust value by framing it within specific constraints of an agent's abilities or behaviors. For example, a plumber may be trustworthy to fix a clogged water drain, but untrustworthy to perform a triple-bypass heart operation, even though both activities deal with improving fluid flow.

4.3 General Structure of Computational Trust Models

The core of the trust problem centers around dealing with the uncertainty of interacting with other agents for some purpose. Hence, computational trust models are designed to give agents the ability to reason about the reciprocity, honesty, and reliability of other agents in order to handle this uncertainty. Since agents in a system are always assumed to have selfish interests, these models take the view point of an agent trying to find the most reliable agents to interact with from a pool of potential agents [112]. Computations generally take into account some combination of the following three components [32]:

- **Experience**. This component is directly measured by an agent, usually as a result of a direct interaction with a neighboring agent.
- **Recommendations**. This component refers to measurements or trust-based information received from a neighboring agent concerning another agent in the network.
- **Knowledge**. At a minimum, this component includes "common knowledge," which implies that every agent in the system definitely knows the truth about some aspect of their existence. However, it can also incorporate any previously evaluated trust values, measurements, or recommendations.

Computational trust models are modeled as a weighted directed graph $\mathcal{G} = (N, E)$, where N is the set of all agents and $E \subseteq N \times N$ is the set of all directed edges between the agents. Each weighted edge represents some trust value from agent $i \in N$ to agent $j \in N$, where $i \neq j$. This trust value can be, for example, a public key certificate (issued by i for j's key), the likelihood of a valid public key certificate, the trustworthiness of j as estimated by i, or some other trust-based measurement.

Computational solutions for dealing with trust-based uncertainty are generally found in the forms of either centralized trust models or decentralized trust models [54]. Centralized trust models assume that at least one "trust agent" is globally available and accessible by all agents in a network. This trust agent may compute the trust values for the entire multi-agent system or help agents in their own trust calculations by providing trust-based information on target agents. The weakness in this type of solution, however, is that the trust agent(s) are single points of failure which can be targeted to massively disrupt the entire trust network. This type of solution also suppresses the subjectivity property of the trust network by assuming that different agents have the same trust-based opinion about the same target.

Decentralized trust models, on the other hand, assume that each agent is the center of their own world and is, therefore, responsible for independently calculating their own trust values for other agents they interact with. This "bottom-up" approach allows for a trust network that is both scalable and fault tolerant. However, the individual agents within the network are potentially more vulnerable to trust-based attacks since it is unlikely that any agent knows the most up-to-date trust values for every other agent in the network. Hence, decentralized trust models often use results from a combination of direct interactions and recommendations about other agents to maintain a reasonably complete picture of the trust network.

4.4 Direct Trust Models

In open systems, where system-wide common goals are difficult to justify, agents develop trust by interacting directly with neighbor agents in order to take advantage of mutual cooperation. It is generally assumed that agents will interact with each other multiple times, thereby making trust an emerging phenomenon. Furthermore, it is also assumed that agents have an incentive to defect, particularly if some agent does not satisfy certain terms in a contract [112].

In early trust work, Sen demonstrated how reciprocity can emerge when agents learn to predict that they will receive future benefits if they cooperate [120]. The prediction is based on a probabilistic decision mechanism that satisfies the set of criteria based on the extra cost incurred by an agent for cooperating. In general, higher costs lower the probability for cooperation. Mukherjee et al. later showed how trust can be acquired if agents know their opponent's chosen move in advance [98]. Agents in a Markov game framework could obtain mutual payoff that, in some cases, is better than the Nash Equilibrium if the agents are allowed to look ahead while selecting actions. In order to guide the agents toward the best non-Nash mutual payoff, the agent would need mutual trust to stick to policies that may deviate from optimal Nash policies. Around the same time, Witkowski et al. proposed a new trust function whereby the trust in an agent is calculated based on their performance in past interactions [147]. The trust function uses two parameters to determine the trust dynamics: the degree to which a positive experience enhances a trust vector element $(0 \leq \alpha \leq 1)$, and the degree to which a negative experience damages the relationship $(0 \leq \beta \leq 1)$. Sabater and Sierra also use a similar idea through the REGRET system, but attribute fuzziness to the notion of performance [118]. The REGRET system also takes into account the social dimension of agents and a hierarchical ontology structure.

4.5 Recommendation Trust Models

Recommendation trust models establish trust on the basis of recommendations alone by seeking reputation information from other agents about third-party agents. Often, reputation exchange is useful to quickly learn about potential trustworthy agents in systems where direct interactions are infrequent or even infeasible. Recommendation trust models often require that at least some agents in a system are able to conduct direct interactions. They also require that the truster is not only able to assess the accuracy of the reputation information, but also the trustworthiness of the agents providing the information.

In earlier work, Abdul-Rahman and Hailes attempted to use social trust characteristics and word-of-mouth to calculate trust in virtual environments [1]. Yu and Singh also developed a method to estimate ratings on a social network through the use of referrals [151]. Both cases show examples of attempts to establish reputation indirectly.

Castelfranchi and Falcone considered subjective perception for reputation establishment and developed socio-cognitive models which incorporate beliefs in competence, willingness, persistence, and motivation [25] [26] [27]. Similarly, Yu and Singh dealt with the absence of information in their reputation model by using the Dempster-Shafter theory to establish belief [150].

In more recent work, Jiang and Baras developed a trust establishment strategy based on local voting for ad hoc networks [69]. In this voting scheme, all of the opinion values from neighbor agents are aggregated to form a trust value. But because a recommender agent may itself be bad, the authors also propose using a confidence value as part of the voting scheme. Theodorakopoulos and Baras extended this work by focusing their research on evaluating trust evidence in ad hoc networks using the theory of semi-rings [137]. The evaluation process was modeled as a shortest path problem on a direct trust game.

4.6 Hybrid Trust Models

Recent work has expanded on earlier formulations of trust models, combining both the direct trust and recommendation models into unified hybrid frameworks. A simple model proposed by Virendra et al. established trust through a linear combination of self-evaluated trust ($T^{(s)}$) and trust evaluated by other nodes ($\boldsymbol{T}^{(o)}$) [142]. Their model calculates the trust that agent i would have for agent j by setting $\boldsymbol{T}_{ij} = \alpha\boldsymbol{T}_{ij}^{(s)} + \beta\boldsymbol{T}_{ij}^{(o)}$, such that $\alpha + \beta = 1$.

In similar work, Fullam and Barber adopted reinforcement learning to learn a parameter ψ that controls how to aggregate information from experience-based and reputation-based trust [46]. The parameter ψ is essentially a weight that regulates how much influence reputation and direct experience have on the final trust value.

Teacy et al. developed a probabilistic trust model called TRAVOS (Trust and Reputation model for Agent-based Virtual OrganizationS) that calculates trust in terms of the confidence that an expected value is within a specific error tolerance [136]. It takes into account past direct interactions, but also factors in reputation information gathered from third parties when personal experience is lacking. The authors showed that TRAVOS can extract a positive influence on performance from reputation, even when more than 50% of the agents are intentionally misleading. That said, TRAVOS assumes that the behavior of agents does not change over time, which is generally not a safe assumption in practice.

Wang and Singh defined trust in terms of belief and certainty [145]. The belief portion of this model is adopted from Jøsang's earlier work, where he defined the trust space as a triple of belief (in a good outcome), disbelief (or belief in a bad outcome), and uncertainty [71]. Wang and Singh, however, updated his ad hoc formulation of certainty, which they derive in terms of evidence based on a statistical measure defined over a probability distribution of positive outcome probabilities. The trust aggregation and concatenation mechanisms are described in terms of a path algebra.

4.7 Trust Dynamics

Trust can change and evolve over time in a multi-agent system on the basis of time, agent experience, and data from other information sources. Ultimately, these changes influence the behavior dynamics of each agent. Trust dynamics are generally characterized by the way trust propagates through a network and the way trust is aggregated with other trust-based information.

Trust propagation refers to the mechanism of distributing trust information throughout a network. It reduces re-computations of trust by other nodes and can be extremely useful in applications that lack infrastructure, autonomy, mobility, and resources. Recommendations are considered the simplest form of trust propagation, generally provided directly from a neighbor agent concerning some other agent in the system. This said, multi-hop, multi-path propagation are also found in the literature. For example, Gray et al. propose a trust propagation method based off the small world phenomena, allowing for an authenticating node to be found in relatively few hops [56]. Ballal and Lewis also discuss the concept of trust consensus for collaborative control and show how the propagation of trust through a network can lead to a global asymptotic trust consensus among all agents [14].

Trust aggregation is the mechanism that combines trust values received from multiple sources or paths about a single agent in a particular context. The purpose of this mechanism is to suppress malicious nodes from altering the correct trust value within the network. Common trust aggregation functions include arithmetic mean, weighted mean, and min-max. However, other methods have been proposed as well. For example, Wang and Singh provide an aggregation method using subjective logic within the context of belief functions [145]. Here, the aggregation updates a trust triplet of belief, disbelief, and uncertainty through evidence summation within a belief function. Bachrach et al. proposed a gossip-based aggregation method called "push-sum," which aggregates rumor values from multiple sources after receiving them a sufficient number of times [10].

Aggregation schemes have turned up in some multi-agent applications. For example, Baras et al. calculate aggregate trust values in autonomous agent networks based on the data flow routes between agents [16]. Also Zhang et al. present a framework to secure data aggregation against false data injection in wireless sensor networks [153]. Their method exploits redundancy in gathered data to evaluate the trustworthiness of each sensor.

Other types of trust dynamics have also been mentioned in the literature, namely trust prediction, trust mirroring, and trust teleportation [24] [61] [130]. **Trust prediction** describes how an agent can determine trust using the predictions of future behaviors (rather than actual observations) as the basis for the trust calculations. **Trust mirroring** uses a truster agent's perceived similarities with another agent as an indicator of future trust. **Trust teleportation** applies trust derived from an existing trust relationship to new relationships that appear to be similar to the existing relationship.

4.8 System-Level Trust

Whereas agent based trust models in previous sections are intended to cultivate trust at the agent-level, protocols of interaction are intended to guarantee trust at the system-level. In short, they are developed to make sure agents will gain some utility if they follow the rules – and lose utility if they don't. Thus, the rules of a system enable agents to trust other agents by the virtue of the different constraints in the system. We briefly describe truth-eliciting, reputation, and security mechanisms for system-level trust in this subsection.

Truth-eliciting protocols force agents to follow the rules by strictly dictating the individual steps in interactions and the information revealed by the agents during those interactions. By doing so, agents should find no better option than to tell the truth. The Vickrey-Clarke-Groves (VCG) mechanism is an example of a truth-eliciting protocol, where agents pay the "damage" they impose on other agents in an auction, thereby ensuring the optimal strategy is to bid the true valuation of an object [108].

Reputation mechanisms force agents to interact with some trust authority to get public ratings on other agents in a system. Zacharia and Maes outlined some basic requirements for practical reputation mechanisms [152].

- It should be costly to change identities.
- New entrants should not be penalized by initially having low reputation.
- Agents with low ratings should be allowed to build up reputation.

- The overhead of performing fake transactions should be high.
- Agents with high reputations should have higher bearing than others on reputation values.
- Agents should be able to provide personalized evaluations.
- Agents should remember reputation values and give more importance to the most recent ones.

Security mechanisms force agents in networks to prove who they say they are. Poslad et al. proposed that identity, access permissions, content integrity, and content privacy are essential for agents to trust each other and their respective messages transmitted across a network [109]. These requirements are specified in the Foundation for Intelligent Physical Agents (FIPA) abstract architecture, and implemented by public key encryption (PGP and X.509) and a certificate infrastructure [68].

Conclusion

To conclude, this chapter provided a high-level overview and history of the computational trust research domain. Its purpose is simply to provide the reader a frame of reference for the work presented in the remaining chapters, and should not be construed as an exhaustive study or survey. Going forward, the reader should keep in mind that computational trust, regardless of how it is calculated, is subjective, asymmetric, and context-dependent; and that it can only be updated through direct experiences or indirect recommendations.

CHAPTER FIVE

ROBOTRUST: A NOVEL COMPUTATIONAL TRUST MODEL

Synopsis

In this chapter, we present a new computational trust model called RoboTrust. This model calculates trustworthiness in agents by determining the smallest value in a set of maximum-likelihood estimates that are based on different historical observations. It has been specifically designed for use in robotics applications; however, its simplicity and compactness lends itself for use in other problem domains as well.

We begin with Section 5.1, which discusses the motivation behind the development of RoboTrust. Section 5.2 presents the theoretical development of RoboTrust for direct trust computation, as well as an extension for the propagation and aggregation of recommendations for indirect trust computation. Section 5.3 demonstrates the behavior of the RoboTrust model under different tolerance and confirmation parameters using observation data generated from a single period of a sine wave. Section 5.4 subjectively compares and contrasts RoboTrust to the work of other researchers. Section 5.5 compares the trust model performance of RoboTrust to two commonly-used probabilistic trust models.

5.1 Overview

In Chapter Four, we provided an overview of the computational trust research domain by presenting a short survey of computational trust characteristics, models, and dynamics, as well as a significant body of computational trust research in MANETs. Despite this, however, there is a notable lack of consideration in the computational trust literature for practical military applications in robotics. This may be due to the way many researchers narrowly define interactions between agents on communication alone, limiting trust model applications to information or transactional systems. It may also be due to the fact that most military robots currently fielded are teleoperated by human operators [88], which reduces the urgency for trust-based soft security relative to more immediate needs [102]. And while there is a growing interest by the U.S. Army to improve solider trust in its robots through increased transparency and meta-cognition, this interest stems from the assertion that "existing robotic systems are notoriously opaque and distrusted" due to their inability to model their own behavior or semantically understand natural human communication [117].

A novel trust model is, therefore, necessary to address the gap in the computational trust literature for military robotics applications. Such a trust model must not only be able to adapt to volatile mission dynamics, but also support a wide range of potential mission profiles, such as bomb disposal, surveillance, reconnaissance, and convoy. In addition, it should be conceptually simple so that it could be easily understood by a mission planner in the field. Furthermore, in order for robots to establish bi-directional trust-based pseudo-relationships with humans, the trust model must also be able to reasonably emulate the biological trust exhibited by both humans and animals.

The trust model proposed in this chapter, named **RoboTrust**, attempts to meet these requirements. We provide a process flow for this model in Figure 5.1, which

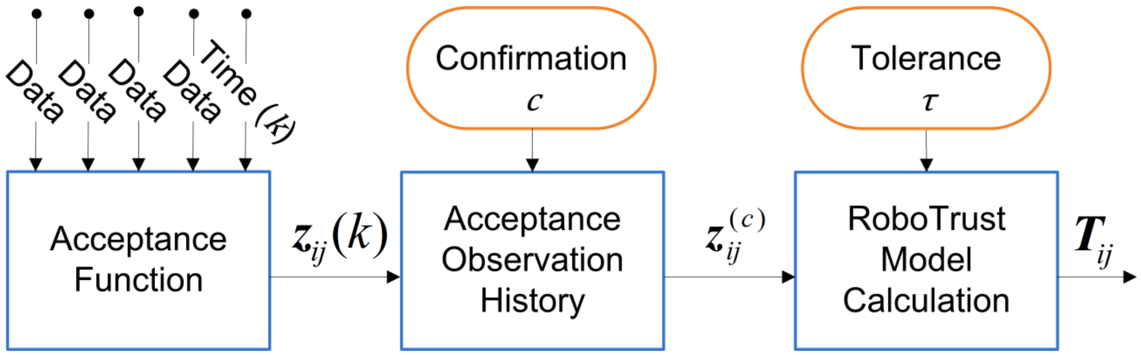


Figure 5.1. Process Flow Chart for the RoboTrust Model.

shows an explicit separation between the context (acceptance function) and the RoboTrust calculation. This separation allows a robotic behavior developer to focus his engineering efforts to precisely and correctly describe contexts without needing to understand how the trust calculation will ultimately evaluate a history of observations. It also allows a mission planner to select predefined contexts from a database and only concentrate on tuning RoboTrust parameters for those selected contexts, per mission requirements. This modularization is a significant advantage for engineering management and platform deployment over other trust models that more tightly couple the context and the trust calculation.

5.2 Theoretical Development

In this section, we describe and derive the RoboTrust model – our novel mechanism to meaningfully cultivate trust towards other agents in multi-agent systems. The proposed trust model establishes context through the use of acceptance functions and interprets context in the RoboTrust calculation through tolerance and confirmation parameters. In addition to showing the derivation of the trust model, we also provide a RoboTrust extension that allows for the consideration of indirect observations about other agents in the network.

5.2.1 The Acceptance Function

The purpose of an acceptance function is to interpret a set of measurements and decide whether or not an agent should collectively deem them as acceptable. In essence, the acceptance function mathematically describes a particular context as a feature space, and then defines the portions of that feature space that the agent should find acceptable.

In our work, we describe an acceptance function z with a binary output that represents either an acceptable (1) or unacceptable (0) result based on a set of observations Z, given by the information function $\rho: k \times F \rightarrow Z$, where $k \in \mathbb{N}$ is a time step, F is the set of feature space attributes, and $Z = \{x | k \in \mathbb{N}, \forall f \in F: x \in \rho(k, f)\}$.

$$z(k): Z(k, F) \rightarrow \{0,1\} \tag{5.1}$$

Note that the explicit mapping of the feature space to the acceptable/unacceptable regions is application-specific and should be defined by the practitioner.

The primary motivation for describing the acceptance function's codomain as a binary set is that the meaning of the output is readily understood as being inside or outside an acceptance region, and does not require any additional interpretation. Technically, though, an acceptance function could also be defined on some continuous range between acceptable and unacceptable extremes – but it would require that the acceptance function knows how to interpolate between each extreme, which can be complicated and somewhat ambiguous in interpretation.

5.2.2 Derivation of the General RoboTrust Model

Let us assume agent i acquires an **acceptance observation history** about agent j from some acceptance function

$$\boldsymbol{z}_{ij} = [\boldsymbol{z}_{ij}(0) \quad \boldsymbol{z}_{ij}(1) \quad \cdots \quad \boldsymbol{z}_{ij}(k)] \tag{5.2}$$

where $\boldsymbol{z}_{ij}(k)$ is the most recent observation. Let us also define two parameters, **tolerance** ($\tau \in \mathbb{N}$) and **confirmation** ($c \in \mathbb{N}$), such that $0 \leq \tau \leq c \leq k$. Now, suppose agent i considers only the $r + 1$ most recent acceptance observations of agent j, where $r \in \mathbb{N}$ and $\tau \leq r \leq c$. Then, let:

$$\boldsymbol{z}_{ij}^{(r)} = [\boldsymbol{z}_{ij}(k-r) \quad \boldsymbol{z}_{ij}(k-r+1) \quad \cdots \quad \boldsymbol{z}_{ij}(k-1) \quad \boldsymbol{z}_{ij}(k)] \tag{5.3}$$

Note that for extreme cases, $\boldsymbol{z}_{ij}^{(0)} = \left[\boldsymbol{z}_{ij}(k)\right]$ and $\boldsymbol{z}_{ij}^{(k)} = \boldsymbol{z}_{ij}$.

Now, let $\boldsymbol{\omega} \in \{0,1\}^{r+1}$ be a sequence of random binary variables associated with these acceptance observations with a discrete probability distribution P that depends on a parameter θ. Then, the likelihood function can be defined as:

$$\boldsymbol{\mathcal{L}}\left(\theta | \boldsymbol{z}_{ij}^{(r)}\right) = P_\theta\left(\boldsymbol{\omega} = \boldsymbol{z}_{ij}^{(r)}\right) \tag{5.4}$$

From the likelihood function, let the parameter θ be the probability that agent i will find the behavior of agent j acceptable with respect to the context defined by some acceptance function. We are specifically interested in the parameter θ that is most likely for a given acceptance observation history $\boldsymbol{z}_{ij}^{(r)}$. However, since there may be different likelihood functions for different acceptance observation histories $\boldsymbol{z}_{ij}^{(r)}$

between the lengths $\tau + 1$ and $c + 1$, we must provide a rule that assigns which likelihood function is most preferred (or which value of r is most preferred). For RoboTrust, we prefer a value of r that provides the most conservative estimate of the most likely parameter θ for a given set of evidence. This can be done by assigning the trust attitude $\boldsymbol{T}_{ij}$ to equal the smallest, most likely probability taken from all acceptance observation histories between the lengths $\tau + 1$ and $c + 1$.

$$\boldsymbol{T}_{ij} = \min \begin{pmatrix} \arg\max\limits_{\theta_\tau} \boldsymbol{\mathcal{L}}\left(\theta_\tau | \boldsymbol{z}_{ij}^{(\tau)}\right) \\ \arg\max\limits_{\theta_{\tau+1}} \boldsymbol{\mathcal{L}}\left(\theta_{\tau+1} | \boldsymbol{z}_{ij}^{(\tau+1)}\right) \\ \vdots \\ \arg\max\limits_{\theta_c} \boldsymbol{\mathcal{L}}\left(\theta_c | \boldsymbol{z}_{ij}^{(c)}\right) \end{pmatrix} \tag{5.5}$$

5.2.3 Specific RoboTrust Model for the Binomial Distribution

Since we assume that the acceptance function codomain is binary, let us suppose that each acceptance observation history is a random variable that comes from a binomial distribution, where there are $\alpha = \sum_{x=0}^{r} \boldsymbol{z}_{ij}(k - x)$ favorable observations from a total of $\beta = r + 1$ most recent observations. Then the likelihood function can be defined as

$$\boldsymbol{\mathcal{L}}(\theta | \alpha, \beta) = \binom{\beta}{\alpha} \theta^{\alpha} (1 - \theta)^{\beta - \alpha} \tag{5.6}$$

We wish to find the maximum likelihood estimate for θ given α and β. This can be done by setting the derivative of the log-likelihood to zero and solving for θ.

$$\log \boldsymbol{\mathcal{L}}(\theta | \alpha, \beta) = \log \binom{\beta}{\alpha} + \alpha \log \theta + (\beta - \alpha) \log(1 - \theta) \tag{5.7}$$

$$\frac{d}{d\theta}\log \boldsymbol{\mathcal{L}}(\theta|\alpha,\beta) = \frac{\alpha}{\theta} - \frac{(\beta-\alpha)}{(1-\theta)} = 0 \tag{5.8}$$

$$\hat{\theta} = \frac{\alpha}{\beta} \tag{5.9}$$

$\hat{\theta}$ denotes the maximum likelihood estimate, which we can incorporate into Equation 5.5 to calculate a trust attitude $\boldsymbol{T}_{ij}$.

$$\boldsymbol{T}_{ij} = \min\left(\begin{matrix} \dfrac{\sum \boldsymbol{z}_{ij}^{(\tau)}}{\tau+1} \\ \dfrac{\sum \boldsymbol{z}_{ij}^{(\tau+1)}}{\tau+2} \\ \vdots \\ \dfrac{\sum \boldsymbol{z}_{ij}^{(c)}}{c+1} \end{matrix}\right) \tag{5.10}$$

5.2.4 Discussion about the RoboTrust Model

One of the key advantages of the RoboTrust model is its simplicity. Besides the acceptance observation history, which is required at some level by all computational trust models, the RoboTrust model requires only two additional inputs, namely the confirmation and tolerance parameters. Both of these inputs are positive whole numbers and provide the necessary information of how observations about a particular context should be interpreted by RoboTrust.

The confirmation parameter (c) controls the growth of trust and establishes the maximum length of an acceptance observation history. We observe from Equation 5.10 that the confirmation parameter also describes the minimum number of consecutive acceptable observations that are necessary to gain complete trust (i.e. $\boldsymbol{T}_{ij} = 1$). Hence, the larger the value of c, the slower the growth of trust.

The tolerance parameter (τ), on the other hand, controls the decay of trust and establishes the minimum number of observations that are required to evaluate trustworthiness. Thus, higher tolerance values put less emphasis on the most current observations since these observations are considered more collectively with older observations. What this means practically is that unacceptable observations are tolerated more when the minimum number of observations is higher (i.e. trust decays slower with higher tolerance). From Equation 5.10, we also see that the tolerance parameter describes the minimum number of consecutive unacceptable observations that are necessary to lose all trust (i.e. $\boldsymbol{T}_{ij} = 0$).

It is important to note a small issue that arises when applying RoboTrust in practice: the question of how to handle initial trust cultivation when the acceptance observation history length is less than $c + 1$ (i.e. $k < c$). While there are several approaches that could resolve this issue, we favor a pessimistic approach that initializes the acceptance observations history with $c + 1$ unfavorable observations and offsets k to equal c. This approach allows agents to assume no trust, which minimizes exposure to trust-based vulnerabilities in initial interactions when trust has not yet been properly cultivated.

5.2.5 RoboTrust Extension for Indirect Trust Aggregation and Propagation

This subsection describes a method to give agents the ability to use RoboTrust to gauge trust about other agents who are not first-neighbors (from a graph theory perspective) and cannot be directly observed. Let us begin by defining the mth-neighbors of i as a recursive set function.

$$J_i^{(m)} = J_i^{(m-1)} \cup N_{J_i^{(m-1)}} \qquad m \geq 1, J_i^{(0)} = \{i\} \tag{5.11}$$

Thus, j is a mth-neighbor of i if $j \in J_i^{(m)} \setminus J_i^{(m-1)}$. Note, however, that this does not imply that i is also a mth-neighbor of j since the network is modeled as a digraph. Furthermore, it may not be possible to indirectly gauge the trust of every mth neighbor in a system since unidirectional relationships in the graph prevent cooperative interactions between any two agents. Therefore, this extension is limited to agents who are the mth-neighbors in bidirectional relationships with i (directly or indirectly).

The RoboTrust model uses direct acceptance observations as its input to calculate trust. For this extension, we use indirectly-acquired acceptance observations from neighbors. These indirect observations can be thought of as "recommendations." Thus, there is no need to change an agent's trust model for a particular context. Rather, an agent relies on the hidden (or private) acceptance functions of its mth-neighbors to determine the acceptance observations. Since there may be multiple hops and paths between two agents in a network, the indirect trust extension is simply a rule for trust information propagation and aggregation. This rule can be stated with the following equation:

$$\boldsymbol{z}_{ij}^{(c)} = \left\lfloor \frac{2\sum_{x \in N_i} \boldsymbol{z}_{xj}^{(c)}}{|N_i| + 1} \right\rfloor \tag{5.12}$$

Equation 5.12 essentially states that if more than half of the indirect acceptance observations at a particular time step are favorable, then agent i can consider his own acceptance observation of agent j at that time step to be favorable; otherwise, his acceptance observation of agent j at that time step is unfavorable. The purpose of the floor function and extra constants in the numerator and denominator is to prevent division-by-zero problems and extraneous logic statements in implementations of this extension.

To show this concept, consider a network of 5 agents, as shown in Figure 5.2. Agent 1 is connected to agent 5 through agent 1's first-neighbors 2, 3, and, 4. Thus, agent 5 is a second-neighbor of agent 1. That said, agent 5 is not the second-neighbors of agent 1 since agent 5 does not have any directed edges to agents 2, 3, and 4.

Suppose agent 1 is interested in indirectly gauging the trust of agent 5. Thus, agent 1 requests the latest 5 acceptance observations about agent 5 from its first-neighbors, namely agents 2, 3, and 4. Each one returns the following observation sequences about agent 5 to agent 1.

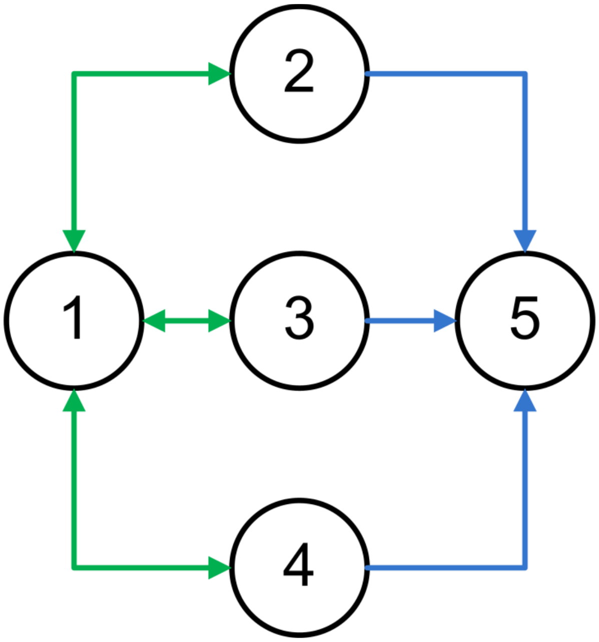


Figure 5.2. Five-Agent Network where Agent 5 is Second-Neighbor of Agent 1.

$$\boldsymbol{z}_{2,5}^{(4)} = [0 \quad 1 \quad 1 \quad 1 \quad 0]$$

$$\boldsymbol{z}_{3,5}^{(4)} = [0 \quad 1 \quad 0 \quad 1 \quad 1]$$

$$\boldsymbol{z}_{4,5}^{(4)} = [0 \quad 0 \quad 1 \quad 1 \quad 1]$$

These sequences can be interpreted as recommendations about agent 5 from agents 2, 3, and 4. Agent 1 can now aggregate these observations using Equation 5.12. In doing so, the indirect observation of agent 1 is

$$\begin{aligned}\boldsymbol{z}_{1,5}^{(4)} &= \left[\left\lfloor\frac{2\times 0}{3+1}\right\rfloor \quad \left\lfloor\frac{2\times 2}{3+1}\right\rfloor \quad \left\lfloor\frac{2\times 2}{3+1}\right\rfloor \quad \left\lfloor\frac{2\times 3}{3+1}\right\rfloor \quad \left\lfloor\frac{2\times 2}{3+1}\right\rfloor\right] \\ &= [0 \quad 1 \quad 1 \quad 1 \quad 1]\end{aligned} \tag{5.13}$$

Agent 1 can now take $\boldsymbol{z}_{1,5}^{(4)}$ and use it in RoboTrust (with its own tolerance and confirmation parameters) as if it directly observed agent 5.

Equation 5.12 can also support trust propagation. Suppose agent 6 enters the network, becomes first-neighbors with agents 1 and 4 (as in Figure 5.3), and wants to indirectly gauge the trust of agent 5. Then, like before, agent 6 requests the latest 5 acceptance observations about node 5 from its first-neighbors.

$$\boldsymbol{z}_{1,5}^{(4)} = [0 \quad 1 \quad 1 \quad 1 \quad 1]$$

$$\boldsymbol{z}_{4,5}^{(4)} = [0 \quad 0 \quad 1 \quad 1 \quad 1]$$

Aggregating these observations with Equation 5.12 results in

$$\begin{aligned}\boldsymbol{z}_{6,5}^{(4)} &= \left[\left\lfloor\frac{2\times 0}{2+1}\right\rfloor \quad \left\lfloor\frac{2\times 1}{2+1}\right\rfloor \quad \left\lfloor\frac{2\times 2}{2+1}\right\rfloor \quad \left\lfloor\frac{2\times 2}{2+1}\right\rfloor \quad \left\lfloor\frac{2\times 2}{2+1}\right\rfloor\right] \\ &= [0 \quad 0 \quad 1 \quad 1 \quad 1]\end{aligned} \tag{5.14}$$

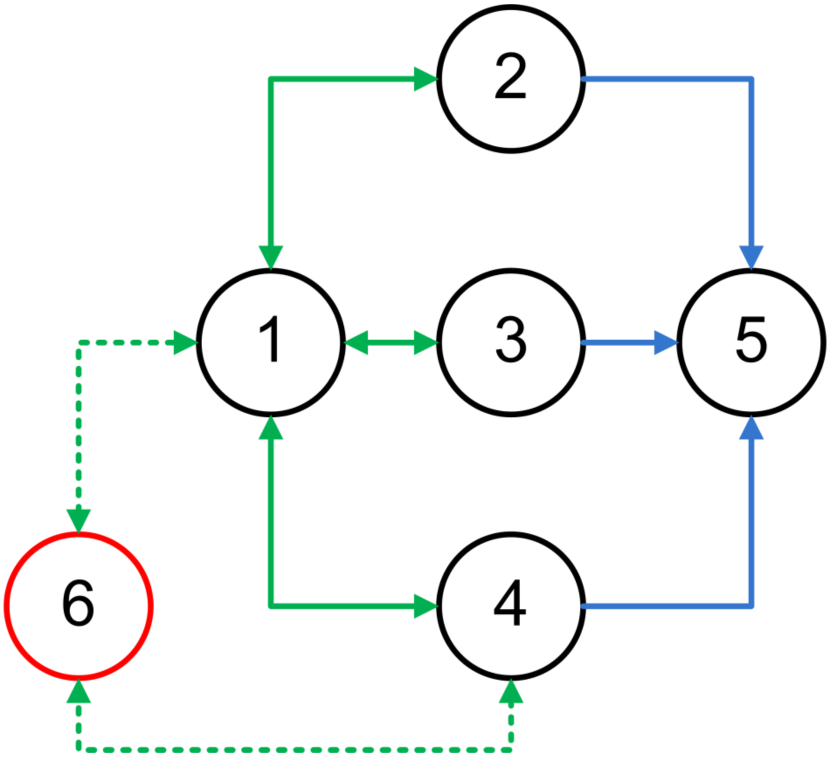


Figure 5.3. Six-Agent network where Agent 5 is Both Second-Neighbor and Third-Neighbor of Agent 6.

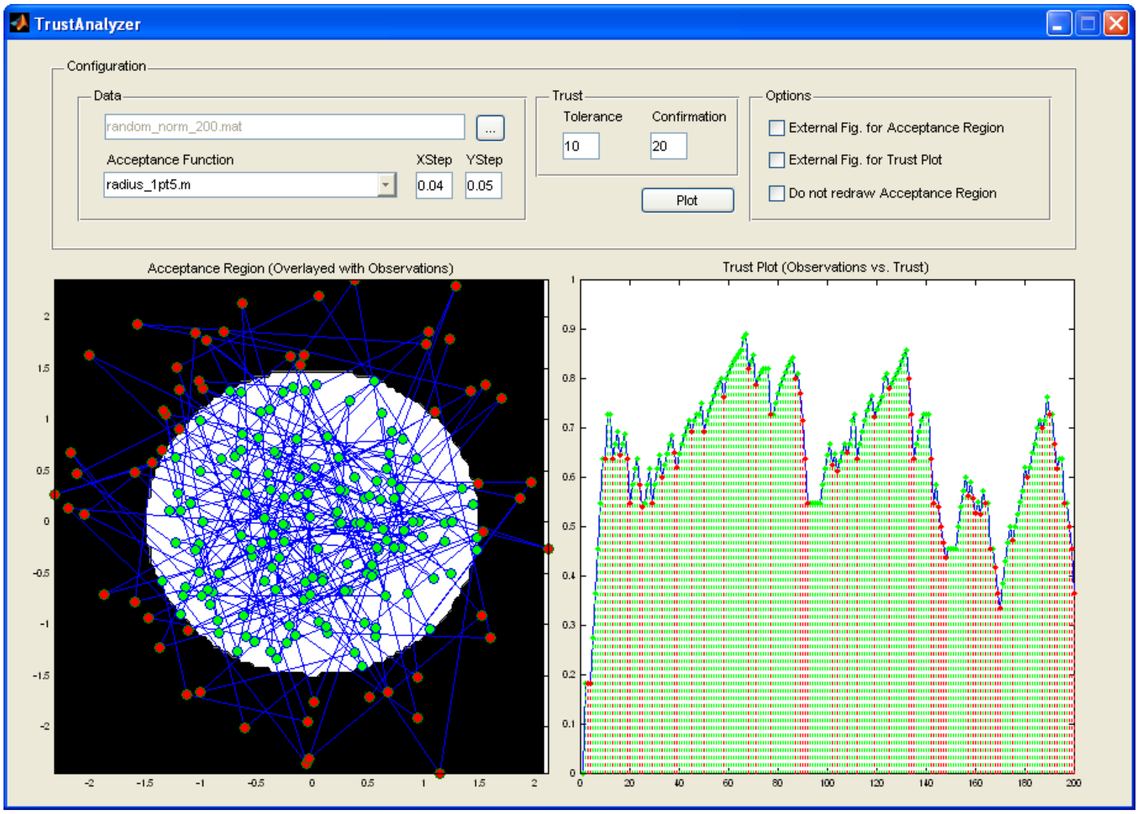
Now, agent 6 can use $\boldsymbol{z}_{6,5}^{(4)}$ in its own RoboTrust model to calculate the trust for agent 5. Thus, we see that trust information about agent 5 propagated to agent 6 from as far as its second-neighbors.

While the examples above illustrate ideal conditions, in practice, additional logic may be necessary to handle situations where agents are unresponsive or do not have acceptance observation data available. An agent may also consider filtering recommendations about other agents from first-neighbors based on the existing trust values of its first-neighbors. These are all application-specific conditions and are beyond the scope of this chapter.

5.3 Simple RoboTrust Demonstration: Sine Wave

In this section, we demonstrate the behavior of the RoboTrust algorithm under different tolerance and confirmation parameters using observation data generated from a single period of a sine wave. Our intention is to provide the reader with an intuitive understanding of the RoboTrust algorithm behavior.

The analysis in this section was performed using a custom Matlab application named "TrustAnalyzer" (Figure 5.4), which allows a user to visually analyze the trust dynamics of 2-dimensional datasets within an acceptance region. TrustAnalyzer



TrustAnalyzer

Configuration

Data: random_norm_200.mat

Acceptance Function: radius_1pt5.m

XStep: 0.04 YStep: 0.05

Trust

Tolerance: 10

Confirmation: 20

Options

□ External Fig. for Acceptance Region

□ External Fig. for Trust Plot

□ Do not redraw Acceptance Region

Figure 5.4. A Screenshot of the Matlab TrustAnalyzer Application.

generates two plots: the Acceptance Region plot and the Trust Plot. The Acceptance Region plot partitions a 2-dimensional feature space according to the selected acceptance function. Acceptable regions are shown in white and unacceptable regions are shown in black. The 2-dimensional dataset of interest also overlays these regions. A data point which lies within the bounds of an acceptable region is marked in green; otherwise, it is marked in red. Blue lines connect each data point to the previous and next data points as indexed in the data set. The Trust Plot shows the trust value after the determination of acceptance of a particular data point. The horizontal axis tracks the index of a data point while the vertical axis tracks the trust value as determined by RoboTrust. As in the Acceptance Region plot, a data point at a particular indexed value that is found to be acceptable is marked in green; otherwise, it is marked in red.

In Figure 5.5, we show the data points from a sine wave overlaid on top of an acceptance region defined by $y \leq |0.6|$. The data shows 7 accepted data points, followed by 18 unaccepted data points, followed by 13 accepted data point, followed by 19 unaccepted data points, followed by 6 accepted data points. Figure 5.6 describes the trust dynamics of these observations with respect to different (τ, c) pairs. The results confirm that as c increases, the rate of change for the trust value in the positive direction changes more slowly. Similarly, the results confirm that as τ increases, the rate of change for the trust value in the negative direction changes more slowly.

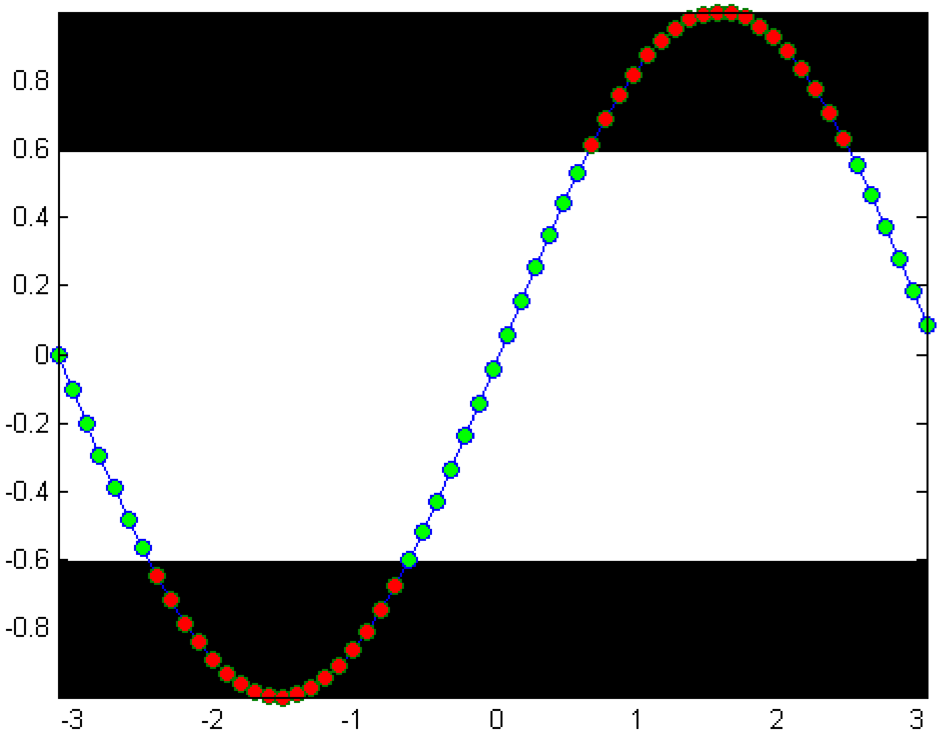
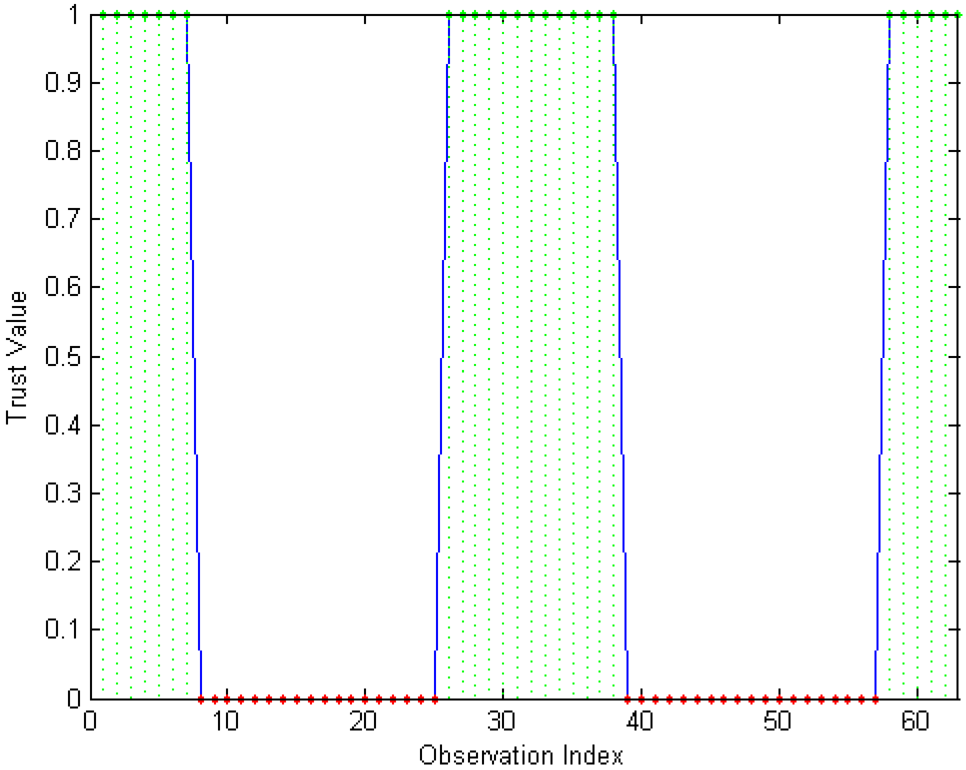


Figure 5.5. Sine Wave with Period $[-\pi, \pi]$ Overlaid on an Acceptance Region Defined by $y \leq |0.6|$. (This figure is presented in color; the black and white reproduction may not be an accurate representation.)

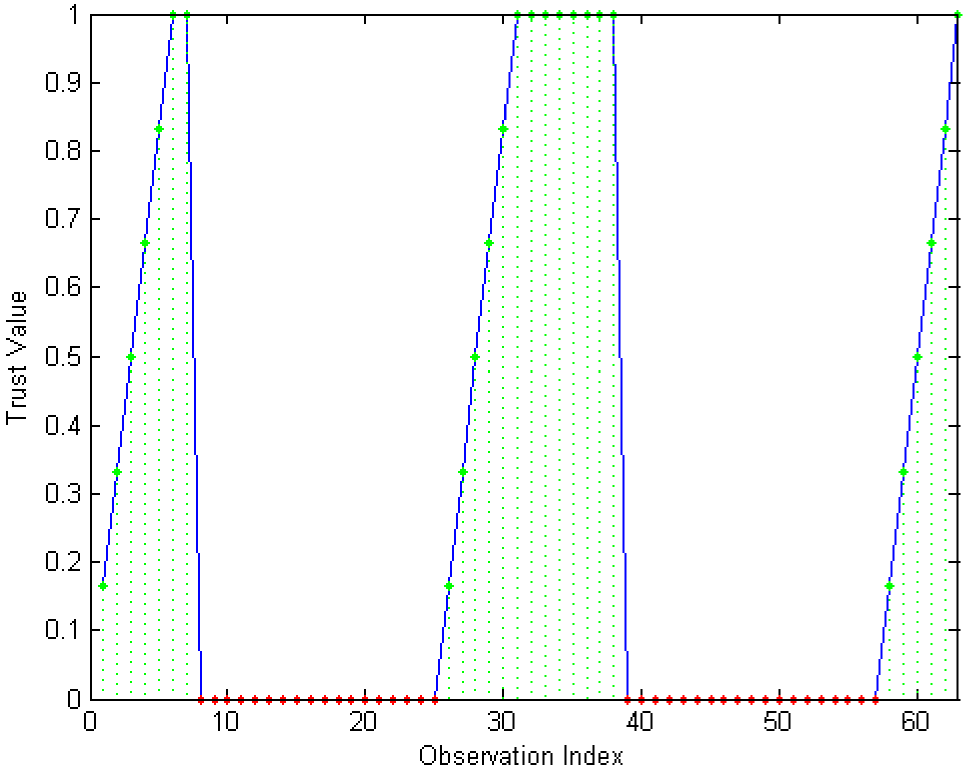
$(\tau, c) = (0, 0)$



(a)

Figure 5.6. Trust Dynamics Results with Different Tolerance and Confirmation Pairs. Provided are the trust dynamics results for: $(\tau, c) = (0, 0)$ (a); $(\tau, c) = (0, 5)$ (b); $(\tau, c) = (5, 5)$ (c); . $(\tau, c) = (0, 20)$ (d); $(\tau, c) = (5, 20)$ (e); $(\tau, c) = (15, 20)$ (f); $(\tau, c) = (0, 40)$ (g); $(\tau, c) = (5, 40)$ (h); $(\tau, c) = (15, 40)$ (i).

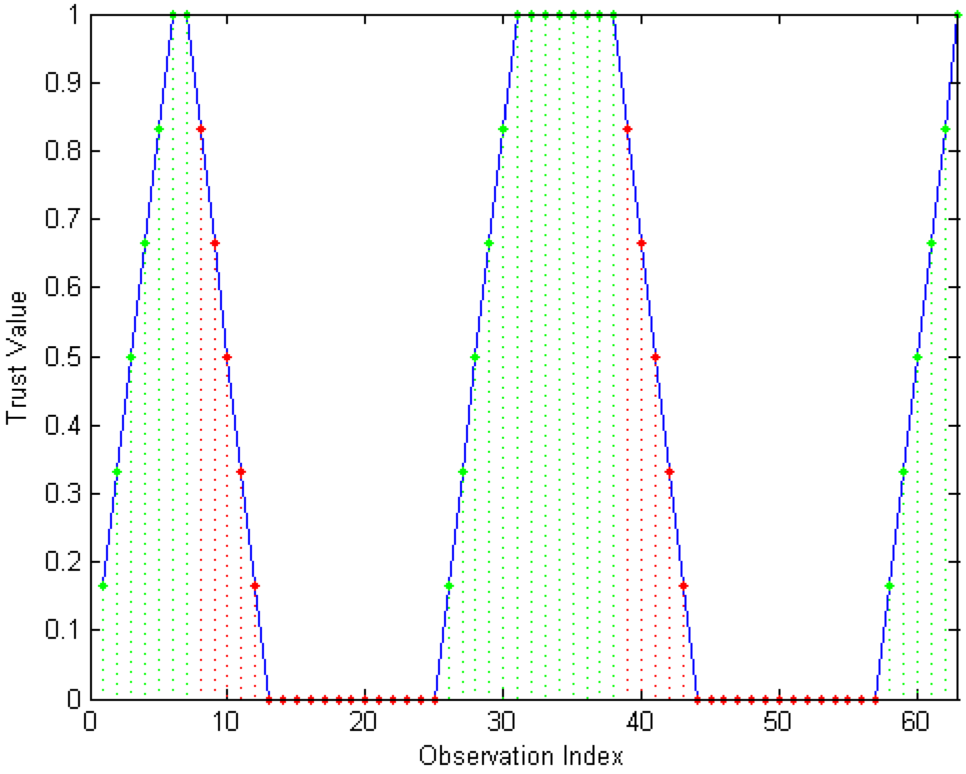
$(\tau, c) = (0, 5)$



(b)

Figure 5.6 – Continued

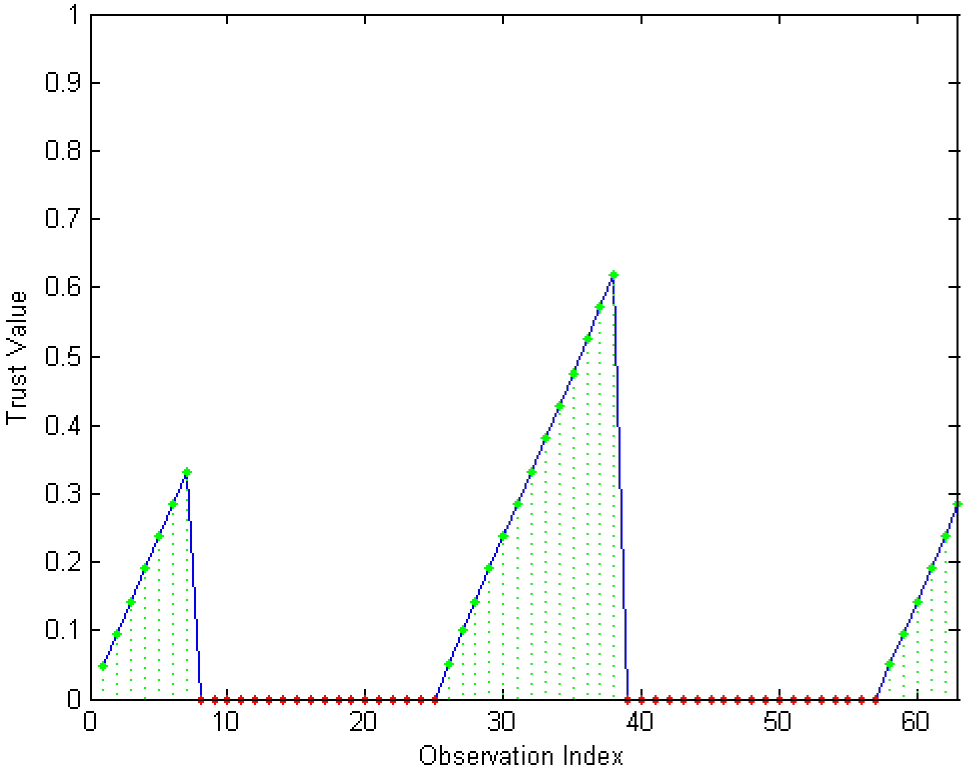
$(\tau, c) = (5, 5)$



(c)

Figure 5.6 – Continued

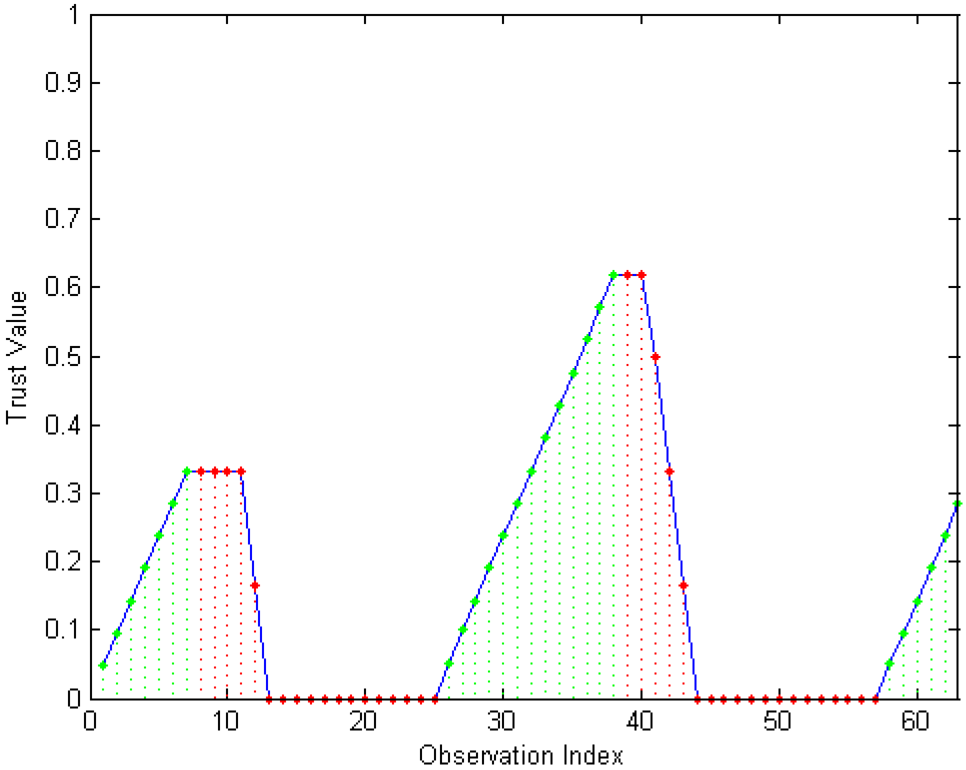
$(\tau, c) = (0, 20)$



(d)

Figure 5.6 – Continued

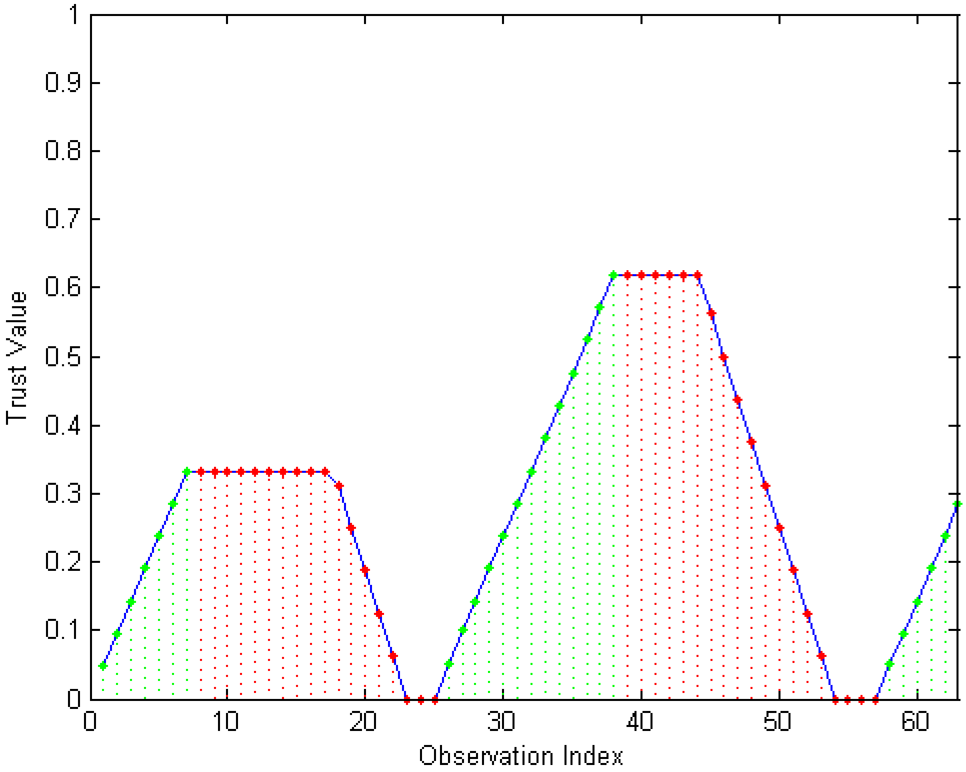
$(\tau, c) = (5, 20)$



(e)

Figure 5.6 – Continued

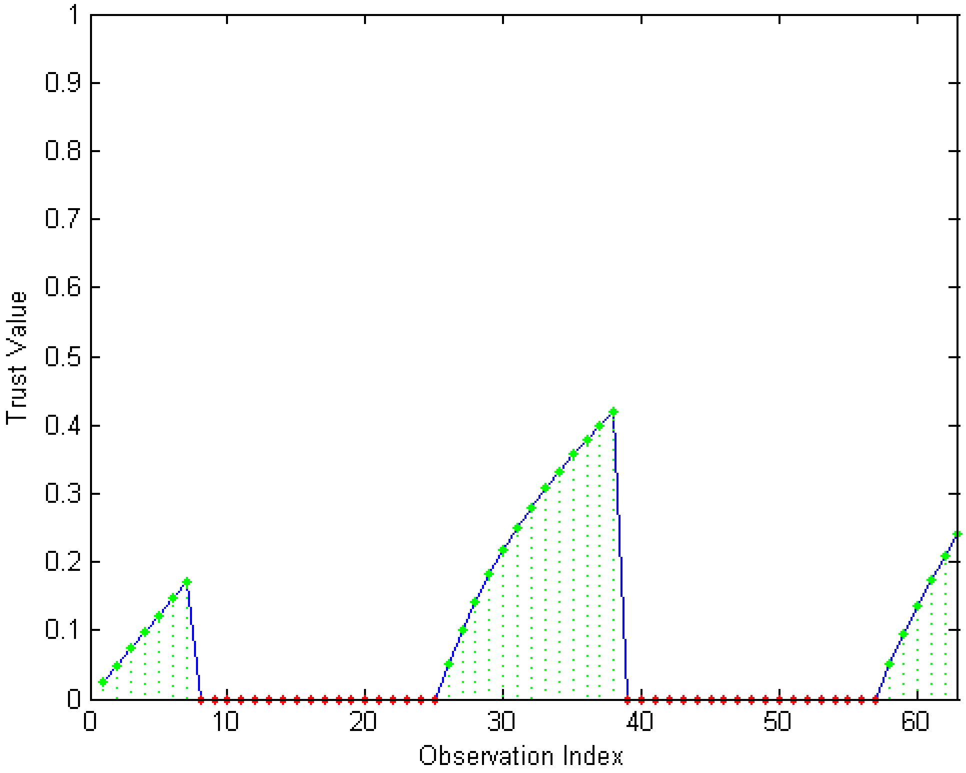
$(\tau, c) = (15, 20)$



(f)

Figure 5.6 – Continued

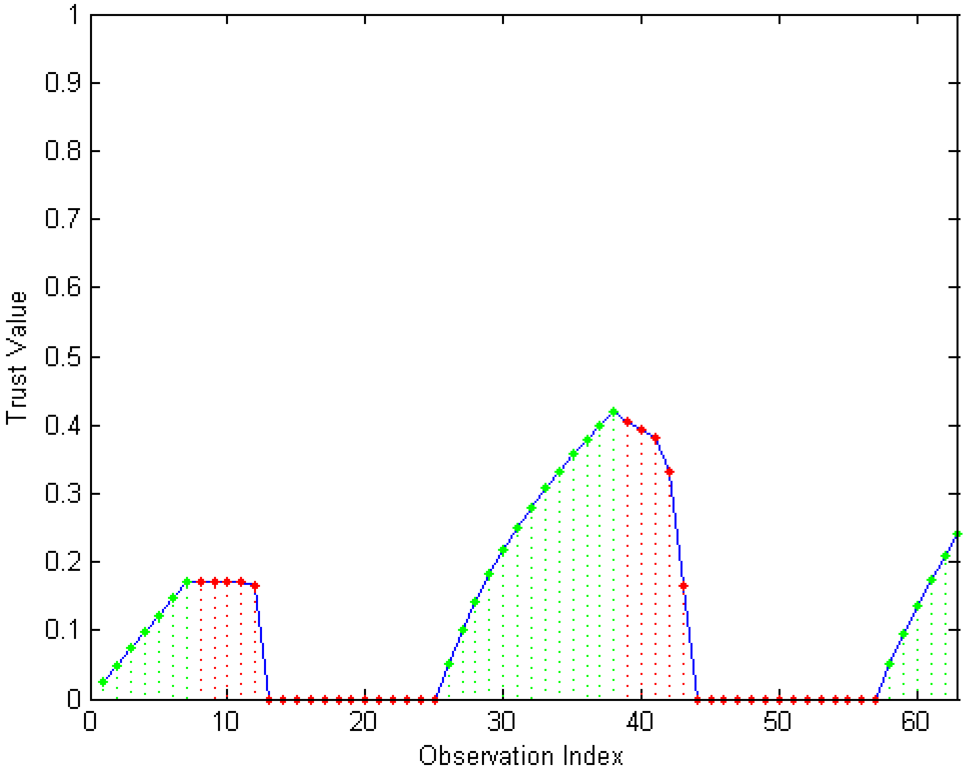
$$(\tau, c) = (0, 40)$$



(g)

Figure 5.6 – Continued

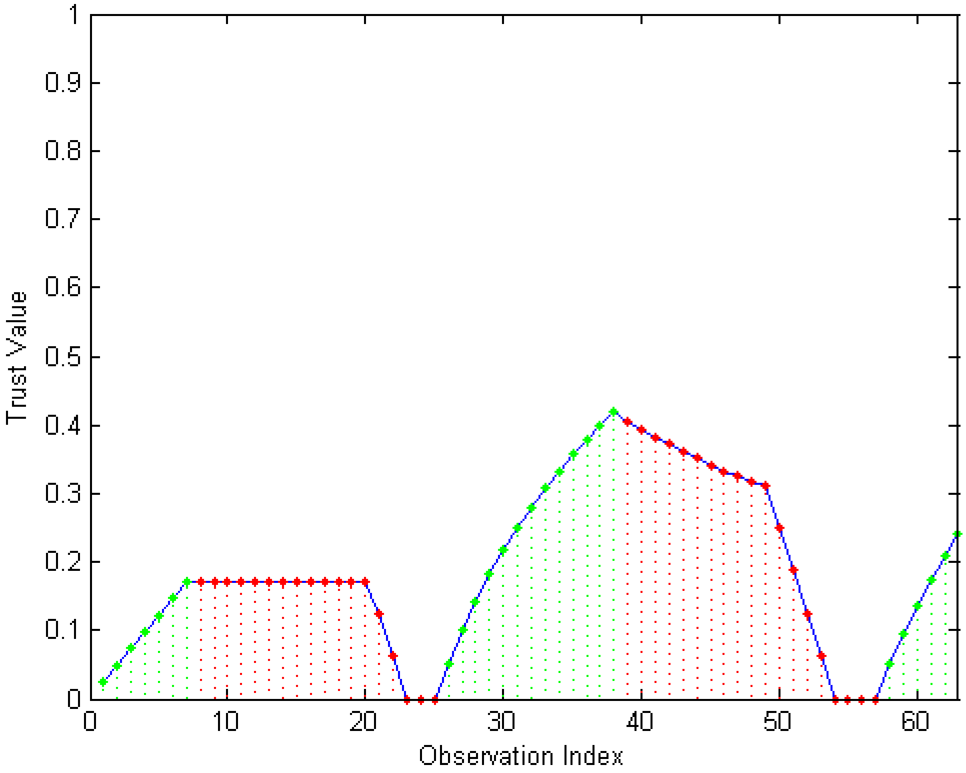
$(\tau, c) = (5, 40)$



(h)

Figure 5.6 – Continued

$(\tau, c) = (15, 40)$



(i)

Figure 5.6 – Continued

5.4 Related Work

The idea of using a probabilistic trust model with a probabilistic trust metric is not new [54]. These metrics are often preferred when **simplicity** is a key criterion within a particular application. However, probabilistic trust models usually lack support for subjectivity, which is used to incorporate an agent's private biases within a trust calculation. That is why more recent computational trust models tend to be multi-faceted in representation [74] [115] [143] [144]. By allowing a higher number of dimensions within a trust opinion, a practitioner is able to more accurately describe a trust opinion and more finely compare and contrast the "value" of different trust opinions. But this additional accuracy comes at the cost of additional complexity, which is often expressed in terms of the additional trust model parameters needed to establish a trust opinion. These parameters also tend to be unintuitive for those outside of Academia, which may cause confusion at best and security holes at worst.

The RoboTrust model balances the benefits of both a simple probabilistic trust metric and a mult-faceted trust representation. The probability assigned to the trust value depends not only on the evidence from an acceptance observation history, but also on the subjective configuration of two whole number parameters: tolerance and confirmation. Furthermore, these two parameters can be easily understood and configured by people without a mathematics or science background, like many in our target warfighter user group. In general, this key advantage can be promoted within any user group to facilitate the adoption of the trust model.

The RoboTrust extension for indirect trust aggregation also differs from other aggregation mechanisms in the literature. Many mechanisms generally combine trust opinions. For example, Jøsang uses discounting and consensus operators on trust opinions for aggregation [74]. CertainTrust uses logical operators (AND, OR, NOT) [99]. RLM implements a reputation feedback aggregator that uses the Kalman method [144]. But the RoboTrust extension uses Equation 5.12 to combine acceptance observation histories rather than trust opinions. By doing so, an agent's trust opinions can remain private, which may be preferred in military multi-agent networks. Also, there is less risk that another agent's subjective interpretation will be mixed with its observable evidence, allowing for more consistency when combining distributed evidence.

5.5 RoboTrust Performance Evaluations

While some have attempted to develop analytical trust model evaluation methods [69] [134], currently, there are no commonly accepted trust model evaluation benchmarks. As such, most researchers use custom simulations to evaluate and compare trust models under specific scenarios. However, these may have limited value since the specific simulations may not be representative of more general conditions. Our evaluations attempt to better balance the benefits of specific value and generality.

5.5.1 Methodology

Our performance evaluations attempt to empirically consider general faults that may occur during interactions between agents. These fault conditions are represented in terms of acceptance observation histories that can be constructed from the results of a wide range of acceptance functions for different contexts. In our evaluations, we consider three types of faults.

- **Persistent Fault**. This fault is characterized by a history of consecutive acceptable observations followed immediately by a history of consecutive unacceptable observations.
- **Periodic Fault**. This fault is characterized by periodically alternating acceptable and unacceptable observation histories for a number of cycles. The period is defined by two components: the number of consecutive acceptable observations and the number of consecutive unacceptable observations.
- **Intermittent Fault**. This fault is characterized by non-periodically alternating acceptable and unacceptable observation histories for a number of cycles. The numbers of consecutive acceptable and unacceptable observations for each cycle are selected randomly from a uniform distribution of natural numbers that are constrained by a minimum and maximum value.

Using an acceptance observation history, $\boldsymbol{z}_{i0}$, as an encoded fault type for input, each agent i determines a probabilistic trust value for test agent 0 at each time step. Then, each agent uses the trust value to classify whether or not the next acceptance observation will be acceptable. We describe this classifier in Equation 5.15.

$$\hat{\boldsymbol{z}}_{i0}(k+1) = \begin{cases} 1 & \boldsymbol{T}_{i0}^{(*)}(k) > 0.5 \\ 0 & \boldsymbol{T}_{i0}^{(*)}(k) \leq 0.5 \end{cases} \tag{5.15}$$

In all of our evaluations, we allow each agent to observe the first twenty-four acceptance observations prior to making its first prediction at $k = 25$. An initial training set like this ensures that each agent has a period of time for trust cultivation.

5.5.2 Metrics

We determine our classification metrics by comparing the classifier value to the true value in the acceptance observation history at a given time step. This can be done efficiently with an XNOR operation (logical equality) between $\boldsymbol{z}_{i0}$ and $\hat{\boldsymbol{z}}_{i0}$. The resultant vector can then be compared against $\boldsymbol{z}_{i0}$ to count the number of true positives, true negatives, false positives, and false negatives for a particular test. We can then apply these counts to the following performance metrics to analyze the predictive performance of the trust-based classifier using a particular trust model.

- **Accuracy**: the proportion of correct classifier predictions over all classifier predictions.

$$Accuracy = \frac{True\ Positives + True\ Negatives}{Total\ Number\ of\ Predictions} \tag{5.16}$$

- **Precision**: the proportion of correct classifier predictions over all predicted class output cases.

$$Precision = \frac{True\ Positives}{True\ Positives + False\ Positives} \tag{5.17}$$

- **Recall**: the proportion of correct classifier predictions over all true class output cases.

$$Recall = \frac{True\ Positives}{True\ Positives + False\ Negatives} \tag{5.18}$$

- **F-Measure**: the weighted harmonic mean of the precision and recall.

$$f^{(n)} = \frac{(n^2 + 1) \times Precision \times Recall}{n^2 \times Precision + Recall} \tag{5.19}$$

We use $f^{(1)}$, which weights the precision and recall evenly.

5.5.3 Competing Trust Models

RoboTrust is a parameterized trust model. Therefore, its performance will vary depending on the values of its parameters. As such, we must evaluate several versions of RoboTrust with different parameters to gauge its expected performance for different types of faults. We selected the following six (τ, c) combinations for our RoboTrust evaluations: $(0,0)$, $(1,2)$, $(1,5)$, $(3,5)$, $(5,8)$, and $(5,10)$. Note that $(0,0)$ implements the "tit-for-tat" strategy, first introduced by Anatol Rapoport in Robert Axelrod's Prisoners' Dilemma tournaments [9].

We evaluated the various RoboTrust models against two probabilistic models commonly used in the computational trust literature for trust value establishment: the expectation of the beta distribution with an ignorance assumption [97] and Bayes' Rule, described in Equations 5.20 and 5.21, respectively.

$$\boldsymbol{T}_{ij}^{(Beta)} = \frac{\left(\sum \boldsymbol{z}_{ij}^{(k)}\right) + 1}{k + 2} \tag{5.20}$$

$$\begin{aligned} \boldsymbol{T}_{ij}^{(Bayes)} &= P\left(\boldsymbol{z}_{ij}(k+1) = 1 | \boldsymbol{z}_{ij}^{(0)}\right) \\ &= \frac{P\left(\boldsymbol{z}_{ij}^{(0)} | \boldsymbol{z}_{ij}(k+1) = 1\right) P\left(\boldsymbol{z}_{ij}(k+1) = 1\right)}{P\left(\boldsymbol{z}_{ij}^{(0)}\right)} \end{aligned} \tag{5.21}$$

5.5.4 Evaluation Tests and Results

We considered a set of twelve evaluation tests for the trust model comparison, which includes one persistent fault, four periodic faults, and seven intermittent faults. Results from these tests are summarized and presented in Tables 5.1 through 5.7.

Table 5.1.

Trust Evaluation Results for Prediction Accuracy

#	Test	Beta Expectation	Bayes’ Rule	RoboTrust (0,0)	RoboTrust (1,2)	RoboTrust (1,5)	RoboTrust (3,5)	RoboTrust (5,8)	RoboTrust (5,10)
1	Persistent Fault (50, 50) : 1 cycle	0.3421	0.9737	**0.9868**	**0.9868**	**0.9868**	0.9737	0.9605	0.9605
2	Periodic Fault (1,1) : 100 cycles	0	**0.5**	0	**0.5**	**0.5**	**0.5**	**0.5**	**0.5**
3	Periodic Fault (1,5) : 100 cycles	**0.8333**	**0.8333**	0.6667	**0.8333**	**0.8333**	**0.8333**	**0.8333**	**0.8333**
4	Periodic Fault (5,5) : 100 cycles	0.3986	0.501	**0.8002**	0.7008	0.502	0.4016	0.2008	0.502
5	Periodic Fault (10,5) : 100 cycles	0.6646	**0.8665**	**0.8665**	0.8001	0.6673	0.6003	0.4668	0.4004
6	Intermittent Fault ([1, 2], [1, 2]) 100 cycles	**0.4963**	0.4926	0.3309	0.3199	0.4081	0.4191	0.4301	0.4338
7	Intermittent Fault ([1, 5], [1, 5]) 100 cycles	0.4581	0.4974	**0.6718**	0.559	0.4855	0.4308	0.4479	0.4513
8	Intermittent Fault ([1, 10], [1, 10]) 100 cycles	0.4955	0.5081	**0.8247**	0.7513	0.6413	0.5805	0.5224	0.5233
9	Intermittent Fault ([5, 10], [5, 10]) 100 cycles	0.4926	0.4933	**0.8688**	0.8032	0.672	0.6064	0.4746	0.4357
10	Intermittent Fault ([10, 15], [3, 5]) 100 cycles	0.7551	**0.8785**	**0.8785**	0.818	0.6971	0.636	0.5275	0.5139
11	Intermittent Fault ([10, 15], [10, 15]) 100 cycles	0.4789	0.5158	**0.9202**	0.8802	0.8	0.7599	0.6798	0.6397
12	Intermittent Fault ([1, 30], [1, 30]) 100 cycles	0.5396	0.5506	**0.9323**	0.8999	0.8452	0.8172	0.771	0.7485

Table 5.2.

Trust Evaluation Results for the Precision Associated with Predicting "0"

#	**Test**	**Beta Expectation**	**Bayes' Rule**	**RoboTrust (0,0)**	**RoboTrust (1,2)**	**RoboTrust (1,5)**	**RoboTrust (3,5)**	**RoboTrust (5,8)**	**RoboTrust (5,10)**
1	Persistent Fault (50, 50) : 1 cycle	NaN	**1**	**1**	**1**	**1**	**1**	**1**	**1**
2	Periodic Fault (1,1) : 100 cycles	0	**0.5**	0	**0.5**	**0.5**	**0.5**	**0.5**	**0.5**
3	Periodic Fault (1,5) : 100 cycles	**0.8333**	**0.8333**	0.8	**0.8333**	**0.8333**	**0.8333**	**0.8333**	**0.8333**
4	Periodic Fault (5,5) : 100 cycles	0	0.5015	**0.8016**	0.6689	0.5026	0.4311	0.2874	0.502
5	Periodic Fault (10,5) : 100 cycles	NaN	**0.8016**	**0.8016**	0.6689	0.5025	0.4311	0.2878	0.2519
6	Intermittent Fault ([1, 2], [1, 2]) 100 cycles	0	**0.4926**	0.3209	0.3867	0.4458	0.4487	0.4521	0.4558
7	Intermittent Fault ([1, 5], [1, 5]) 100 cycles	0.4618	0.4974	**0.6701**	0.5426	0.49	0.4537	0.4638	0.4685
8	Intermittent Fault ([1, 10], [1, 10]) 100 cycles	0.5143	0.5077	**0.8272**	0.7173	0.6012	0.5659	0.5224	0.521
9	Intermittent Fault ([5, 10], [5, 10]) 100 cycles	0.4043	0.4943	**0.8676**	0.7661	0.6209	0.5812	0.4759	0.4511
10	Intermittent Fault ([10, 15], [3, 5]) 100 cycles	NaN	**0.7525**	**0.7525**	0.6032	0.4319	0.3367	0.1757	0.1692
11	Intermittent Fault ([10, 15], [10, 15]) 100 cycles	0.4657	0.5093	**0.921**	0.8529	0.743	0.7253	0.6565	0.6142
12	Intermittent Fault ([1, 30], [1, 30]) 100 cycles	0.5481	0.5479	**0.9379**	0.8842	0.8042	0.7959	0.7585	0.73

Table 5.3.

Trust Evaluation Results for the Precision Associated with Predicting "1"

#	Test	Beta Expectation	Bayes' Rule	RoboTrust (0,0)	RoboTrust (1,2)	RoboTrust (1,5)	RoboTrust (3,5)	RoboTrust (5,8)	RoboTrust (5,10)
1	Persistent Fault (50, 50) : 1 cycle	0.3421	0.9286	**0.963**	**0.963**	**0.963**	0.9286	0.8966	0.8966
2	Periodic Fault (1,1) : 100 cycles	0	NaN	0	NaN	NaN	NaN	NaN	NaN
3	Periodic Fault (1,5) : 100 cycles	NaN	NaN	0	NaN	NaN	NaN	NaN	NaN
4	Periodic Fault (5,5) : 100 cycles	0.4425	0	**0.7988**	0.7487	0.5	0.3333	0	NaN
5	Periodic Fault (10,5) : 100 cycles	0.6646	**0.8992**	**0.8992**	0.888	0.8561	0.7484	0.6231	0.5696
6	Intermittent Fault ([1, 2], [1, 2]) 100 cycles	**0.5019**	NaN	0.3406	0	0	0.2368	0.3396	0.3261
7	Intermittent Fault ([1, 5], [1, 5]) 100 cycles	0.4531	NaN	**0.6735**	0.5909	0.4588	0.3511	0.3986	0.3761
8	Intermittent Fault ([1, 10], [1, 10]) 100 cycles	0.4936	0.5714	**0.8221**	0.8013	0.7517	0.6096	0.5222	0.5292
9	Intermittent Fault ([5, 10], [5, 10]) 100 cycles	0.4986	0.2	**0.87**	0.8506	0.787	0.6487	0.4723	0.3991
10	Intermittent Fault ([10, 15], [3, 5]) 100 cycles	0.7551	**0.9192**	**0.9192**	0.9122	0.8937	0.8078	0.7177	0.7117
11	Intermittent Fault ([10, 15], [10, 15]) 100 cycles	0.4839	0.8667	**0.9195**	0.9125	0.8939	0.8081	0.7122	0.6817
12	Intermittent Fault ([1, 30], [1, 30]) 100 cycles	0.4811	0.8462	**0.9257**	0.9214	0.9183	0.8506	0.7905	0.7808

Table 5.4.

Trust Evaluation Results for the Recall Associated with Predicting "0"

#	Test	Beta Expectation	Bayes' Rule	RoboTrust (0,0)	RoboTrust (1,2)	RoboTrust (1,5)	RoboTrust (3,5)	RoboTrust (5,8)	RoboTrust (5,10)
1	Persistent Fault (50, 50) : 1 cycle	0	0.96	**0.98**	**0.98**	**0.98**	0.96	0.94	0.94
2	Periodic Fault (1,1) : 100 cycles	0	**1**	0	**1**	**1**	**1**	**1**	**1**
3	Periodic Fault (1,5) : 100 cycles	**1**	**1**	0.8	**1**	**1**	**1**	**1**	**1**
4	Periodic Fault (5,5) : 100 cycles	0	0.998	0.8	0.8	0.8	0.6	0.4	**1**
5	Periodic Fault (10,5) : 100 cycles	0	**0.8**	**0.8**	**0.8**	**0.8**	0.6	0.4	0.4
6	Intermittent Fault ([1, 2], [1, 2]) 100 cycles	0	**1**	0.3209	0.6493	0.8284	0.7836	0.7388	0.7687
7	Intermittent Fault ([1, 5], [1, 5]) 100 cycles	0.5395	**1**	0.6701	0.7216	0.8419	0.7079	0.7045	0.7663
8	Intermittent Fault ([1, 10], [1, 10]) 100 cycles	0.0952	**0.9947**	0.8272	0.8413	0.8695	0.7425	0.6772	0.7443
9	Intermittent Fault ([5, 10], [5, 10]) 100 cycles	0.0514	**0.9946**	0.8676	0.8676	0.8676	0.7351	0.6014	0.6419
10	Intermittent Fault ([10, 15], [3, 5]) 100 cycles	0	**0.7506**	**0.7506**	**0.7506**	**0.7506**	0.5013	0.2519	0.2519
11	Intermittent Fault ([10, 15], [10, 15]) 100 cycles	0.2514	**0.9952**	0.9202	0.9202	0.9202	0.8405	0.7607	0.7607
12	Intermittent Fault ([1, 30], [1, 30]) 100 cycles	0.8795	**0.9975**	0.9379	0.9391	0.946	0.8933	0.85	0.8537

Table 5.5.

Trust Evaluation Results for the Recall Associated with Predicting “1”

#	Test	Beta Expectation	Bayes’ Rule	RoboTrust (0,0)	RoboTrust (1,2)	RoboTrust (1,5)	RoboTrust (3,5)	RoboTrust (5,8)	RoboTrust (5,10)
1	Persistent Fault (50, 50) : 1 cycle	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**
2	Periodic Fault (1,1) : 100 cycles	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
3	Periodic Fault (1,5) : 100 cycles	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
4	Periodic Fault (5,5) : 100 cycles	**0.8004**	0	**0.8004**	0.6008	0.2016	0.2016	0	0
5	Periodic Fault (10,5) : 100 cycles	1	**0.9001**	**0.9001**	0.8002	0.6004	0.6004	0.5005	0.4006
6	Intermittent Fault ([1, 2], [1, 2]) 100 cycles	**0.9783**	0	0.3406	0	0	0.0652	0.1304	0.1087
7	Intermittent Fault ([1, 5], [1, 5]) 100 cycles	0.3776	0	**0.6735**	0.398	0.1327	0.1565	0.1939	0.1395
8	Intermittent Fault ([1, 10], [1, 10]) 100 cycles	**0.9074**	0.0073	0.8221	0.6588	0.4065	0.4138	0.363	0.2958
9	Intermittent Fault ([5, 10], [5, 10]) 100 cycles	**0.9257**	0.0013	0.87	0.7401	0.4801	0.4801	0.3501	0.2334
10	Intermittent Fault ([10, 15], [3, 5]) 100 cycles	**1**	0.9199	0.9199	0.8399	0.6797	0.6797	0.6168	0.5989
11	Intermittent Fault ([10, 15], [10, 15]) 100 cycles	0.7087	0.0317	**0.9203**	0.8397	0.6786	0.6786	0.598	0.5175
12	Intermittent Fault ([1, 30], [1, 30]) 100 cycles	0.1335	0.0165	**0.9257**	0.853	0.7247	0.7262	0.6767	0.6227

Table 5.6.

Trust Evaluation Results for the F-Measure Associated with Predicting "0"

#	Test	Beta Expectation	Bayes' Rule	RoboTrust (0,0)	RoboTrust (1,2)	RoboTrust (1,5)	RoboTrust (3,5)	RoboTrust (5,8)	RoboTrust (5,10)
1	Persistent Fault (50, 50) : 1 cycle	NaN	0.9796	**0.9899**	**0.9899**	**0.9899**	0.9796	0.9691	0.9691
2	Periodic Fault (1,1) : 100 cycles	NaN	**0.6667**	NaN	**0.6667**	**0.6667**	**0.6667**	**0.6667**	**0.6667**
3	Periodic Fault (1,5) : 100 cycles	**0.9091**	**0.9091**	0.8	**0.9091**	**0.9091**	**0.9091**	**0.9091**	**0.9091**
4	Periodic Fault (5,5) : 100 cycles	NaN	0.6676	**0.8008**	0.7286	0.6173	0.5017	0.3345	0.6685
5	Periodic Fault (10,5) : 100 cycles	NaN	**0.8008**	**0.8008**	0.7286	0.6173	0.5017	0.3347	0.3091
6	Intermittent Fault ([1, 2], [1, 2]) 100 cycles	NaN	**0.6601**	0.3209	0.4847	0.5796	0.5707	0.5609	0.5722
7	Intermittent Fault ([1, 5], [1, 5]) 100 cycles	0.4976	0.6644	**0.6701**	0.6195	0.6195	0.553	0.5593	0.5815
8	Intermittent Fault ([1, 10], [1, 10]) 100 cycles	0.1607	0.6722	**0.8272**	0.7744	0.7109	0.6423	0.5899	0.6129
9	Intermittent Fault ([5, 10], [5, 10]) 100 cycles	0.0911	0.6604	**0.8676**	0.8137	0.7238	0.6492	0.5313	0.5298
10	Intermittent Fault ([10, 15], [3, 5]) 100 cycles	NaN	**0.7516**	**0.7516**	0.6689	0.5483	0.4028	0.207	0.2024
11	Intermittent Fault ([10, 15], [10, 15]) 100 cycles	0.3265	0.6738	**0.9206**	0.8853	0.8222	0.7786	0.7047	0.6796
12	Intermittent Fault ([1, 30], [1, 30]) 100 cycles	0.6753	0.7073	**0.9379**	0.9108	0.8693	0.8418	0.8017	0.787

Table 5.7.

Trust Evaluation Results for the F-Measure Associated with Predicting "1"

#	Test	Beta Expectation	Bayes' Rule	RoboTrust (0,0)	RoboTrust (1,2)	RoboTrust (1,5)	RoboTrust (3,5)	RoboTrust (5,8)	RoboTrust (5,10)
1	Persistent Fault (50, 50) : 1 cycle	0.5098	0.963	**0.9811**	**0.9811**	**0.9811**	0.963	0.9455	0.9455
2	Periodic Fault (1,1) : 100 cycles	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
3	Periodic Fault (1,5) : 100 cycles	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
4	Periodic Fault (5,5) : 100 cycles	0.57	NaN	**0.7996**	0.6667	0.2874	0.2513	NaN	NaN
5	Periodic Fault (10,5) : 100 cycles	0.7985	**0.8996**	**0.8996**	0.8418	0.7058	0.6663	0.5551	0.4704
6	Intermittent Fault ([1, 2], [1, 2]) 100 cycles	**0.6634**	NaN	0.3406	NaN	NaN	0.1023	0.1885	0.163
7	Intermittent Fault ([1, 5], [1, 5]) 100 cycles	0.4119	NaN	0.6735	0.4756	0.2058	0.2165	0.2609	0.2035
8	Intermittent Fault ([1, 10], [1, 10]) 100 cycles	0.6394	0.0143	0.8221	0.7231	0.5277	0.493	0.4283	0.3795
9	Intermittent Fault ([5, 10], [5, 10]) 100 cycles	0.6481	0.0026	0.87	0.7915	0.5964	0.5518	0.4021	0.2946
10	Intermittent Fault ([10, 15], [3, 5]) 100 cycles	0.8605	0.9196	0.9196	0.8745	0.7722	0.7382	0.6634	0.6504
11	Intermittent Fault ([10, 15], [10, 15]) 100 cycles	0.5751	0.0612	0.9199	0.8746	0.7715	0.7377	0.6502	0.5883
12	Intermittent Fault ([1, 30], [1, 30]) 100 cycles	0.209	0.0324	0.9257	0.8859	0.8101	0.7835	0.7292	0.6928

For the persistent fault test, we evaluated how provokable a trust model is. The test was parameterized to consider 50 acceptable observations followed by 50 unacceptable observations for 1 cycle. Our results for Test #1 show that the Beta Expectation function is the least provokable trust model in our group, taking the longest amount of time to adjust to a series of unacceptable observations following a long history of acceptable observations. Bayes' Rule and RoboTrust $(0,0)$, however, were both shown to be the most provokable, calling for an immediate loss of all trust for a single defection.

For the periodic fault tests, we considered four cases for 100 cycles each, parameterized by the following consecutive acceptable/unacceptable observation pairs: $(1,1)$, $(1,5)$, $(5,5)$, and $(10,5)$. Test #2 evaluated the performance of a trust model when exposed to evidence with a maximum entropy rate of 1 bit per observation. Our results show that both the Beta Expectation and RoboTrust $(0,0)$ achieved an accuracy of 0%, but for different reasons; the Beta Expectation trust value oscillated near the trust threshold while RoboTrust $(0,0)$ oscillated between the trust values 0 and 1. The other trust models achieved an equivalent accuracy of 50%. Analyses of the precision and recall results indicate that the trust classifier for each of these trust models outputted a 0 in every case.

Tests #3, #4, and #5 evaluated each trust model's response to different ratios of consecutive acceptable and unacceptable observation lengths: $(1:5)$, $(5:5)$, and $(10:5)$, respectively. RoboTrust $(0,0)$ performed the worst for Test #3, but outperformed all other trust models for Test #4 and matched the best performance of Bayes' Rule for Test #5. Collectively, though, RoboTrust $(1,2)$ showed the most balanced acceptable performance for all three tests. The results also show that slower trust growth and decay rates negatively impact the accuracy of the RoboTrust classifiers for these tests.

For the intermittent fault tests, we considered seven cases of 100 cycles each. Test #6 considered intermittent high frequency oscillations between acceptable and unacceptable observations. Both Beta Expectation and Bayes' Rule outperformed all RoboTrust models for this case. However, accuracy rates of the RoboTrust models tended to increase in step with slower growth and decay rates. That said, the best accuracy rate for all trust models was still below 50%, indicating that no classifier performed better than a uniform random prediction selection at each time step.

The remaining intermittent fault tests were simply different variations of the intermittent faults intended to explore the overall fault space. On the whole, RoboTrust $(0,0)$ outperformed all other trust models for each of these tests. Also, both Beta Expectation and Bayes' Rule performed about the same at about $50\% \pm 4\%$ accuracy, but in general, worse than the other RoboTrust models. The only exception to this was Test #10, in which both Beta Expectation and Bayes' Rule generally outperformed the RoboTrust models in accuracy.

Like the periodic tests, the results for the intermittent tests show that slower trust growth and decay rates negatively impact the accuracy of the RoboTrust classifiers. However, the results from Test #11 and #12 show that the negative impact can be minimized when there are longer consecutive observations (either acceptable or unacceptable), or more variability with the intermittent fault periods. We expect that, for military scenarios, these kind of intermittent faults will be the dominant case, since sustained high frequency oscillations between acceptable and unacceptable observations will be considered to be too erratic for any potential cooperative gains.

With the incredible results from RoboTrust $(0,0)$, one may question why any agent should consider using any other strategy besides “tit-for-tat”. This is a fair concern on the surface. However, we must remember that the given acceptance observation histories for each fault type were context-free, and therefore, assumed to be always correct. But in practice, observations may not always be correctly perceived.

For example, consider the scenario where the lead vehicle in a convoy abruptly drives off-road during an otherwise routine convoy mission. The leader’s immediate follower vehicle may perceive that behavior as unacceptable upon seeing it. However, if the leader’s intention was to avoid an IED in the road, this abrupt behavior is actually acceptable. Thus, if the follower was using RoboTrust $(0,0)$ as its underlying trust model, it would have likely disengaged from following its leader, which could have resulted in it being hit by the IED. A more tolerant trust model, such as RoboTrust $(5,10)$, would have likely allowed the follower to follow the leader off-road for a temporary amount of time – perhaps long enough to avoid the IED. But if it was the case that the leader made a classification error about the IED, and thereby, incorrectly decided to drive off-road, then the worst result for the follower would have been an unnecessary detour.

As such, trust models can give trusted agents the benefit of the doubt for short periods of time. They can also give untrusted agents the motivation to prove themselves with sustained acceptable behaviors.

Conclusion

In summary, this chapter presents the primary contribution of this work: the RoboTrust model. This model explicitly separates the context of an observation from the actual trust calculation, which provides significant advantages in engineering management and platform deployment for military robot developers. The RoboTrust calculation itself determines the smallest, most likely probability of an acceptable observation taken from various historical perspectives, and uses this value as a measure of trustworthiness. An extension to RoboTrust provided a means to gauge trustworthiness about other agents who are not first-neighbors by relying on the hidden acceptance functions to determine an acceptance observation history. We also provided a toy demonstration of the RoboTrust algorithm under different tolerance and confirmation parameters to provide the reader with an intuitive understanding of the algorithm's behavior. Furthermore, we subjectively and objectively compared and contrasted RoboTrust to other related trust models.

The next two chapters will integrate RoboTrust into two meaningful applications, namely: multi-agent consensus and autonomous convoy soft security. The multi-agent consensus problem represents a traditional multi-agent "controls" application while the autonomous convoy soft security problem represents a traditional multi-agent "decision" application.

CHAPTER SIX

TRUST-BASED CONSENSUS FOR MULTI-AGENT SYSTEMS

Synopsis

In this chapter, we present a high-level overview of the consensus problem. Then, we provide a distributed, discrete-time, trust-based consensus protocol and prove its asymptotic convergence to an agreement space. Finally, we analyze the trust-based consensus algorithm under two overarching conditions: static-trust and dynamic trust using a simple three-agent network.

6.1 Consensus Problem Statement

In multi-agent systems, consensus means that each agent reaches an agreement with all other agents in the network about a particular quantity of interest. This is generally done through a consensus protocol, which defines how agents will interact with each other in order to update their current state. The literature contains extensive work on consensus protocols [30] [113] [131] ; and it has been shown that solutions to the consensus problem have broad applications to multi-agent systems in areas such as cooperative control, formation control, flocking, and sensor fusion [104]. A handful of researchers have incorporated trust into these consensus protocols [14].

Let $\boldsymbol{x}_i \in \mathbb{R}^n$ be the public decision vector for each agent $i \in N$. Consider a network of decision-making agents with dynamics $\dot{\boldsymbol{x}}_{\imath} = \boldsymbol{u}_i$ interested in reaching a consensus via local communication with their neighbors on a graph $\mathcal{G} = (N, E)$, where

N is the set of all agents and $E \subseteq N \times N$ is the set of all directed edges between the agents. Let N_i be the set of first-neighbors of agent i. Consensus, by definition, means that all agents asymptotically converge to a one-dimensional agreement space that is characterized by:

$$\boldsymbol{x}_1 = \boldsymbol{x}_2 = \cdots = \boldsymbol{x}_{|N|} \tag{6.1}$$

In other words, the solution can be described as a vector $\boldsymbol{x} = \alpha\mathbf{1}$, where $\mathbf{1} = [1 \quad 1 \quad \cdots \quad 1]^{\mathrm{T}}$ and α is a scalar real value equal to the final consensus value. It has been shown in the literature that $\dot{\boldsymbol{x}} = -\boldsymbol{L}\boldsymbol{x}$ is a distributed consensus algorithm that solves the consensus problem [104]. Here, $\boldsymbol{L}$ is the Graph Laplacian defined as the difference between the diagonal degree matrix ($\boldsymbol{D}_{|N|\times|N|}$) and the adjacency matrix ($\boldsymbol{A}_{|N|\times|N|}$). The elements of $\boldsymbol{A}$, denoted as $\boldsymbol{A}_{ij}$, represent the number of directed edges from i to j (i.e. $\boldsymbol{A}_{ij} \in \{0, 1\}$) and the diagonal elements of $\boldsymbol{D}$, denoted as $\boldsymbol{D}_{ii}$, are the number of outgoing edges incident to i. Note that $\boldsymbol{D}_{ij} = 0$ for all $i \neq j$.

6.2 Distributed, Discrete-Time, Trust-Based Consensus Protocol

In order to develop any trust-based method, one must first define how to manage trust between each agent in a system. In our protocol, we use the definition from Equation 3.16, where the values of $\boldsymbol{T}_{ij}$ represent the probability that agent j is trustworthy from the perspective of agent i.

We now present the following trust-based consensus protocol for discrete-time agents with dynamics $\boldsymbol{x}_i(k+1) = \boldsymbol{x}_i(k) + \epsilon\boldsymbol{u}_i(k)$ for a fixed graph topology

$$\boldsymbol{u}_i(k) = \Delta_i^{-1} \sum_{j \in N} \boldsymbol{T}_{ij} \boldsymbol{A}_{ij} \big(\boldsymbol{x}_j(k) - \boldsymbol{x}_i(k)\big) \tag{6.2}$$

where $\Delta_i = \sum_{j \in N} \boldsymbol{T}_{ij} \boldsymbol{A}_{ij}$ is the weighted degree of all outgoing edges of i and $0 < \epsilon < 1$ is the step-size.

This protocol can be expressed in matrix form as:

$$\boldsymbol{u}(k) \ = -(\boldsymbol{I} - \boldsymbol{D}^{-1}\boldsymbol{W})\boldsymbol{x}(k) \tag{6.3}$$

where $\boldsymbol{W} = \boldsymbol{T} \circ \boldsymbol{A}$ is the weighted adjacency matrix, such that $\boldsymbol{T}$ is the trust matrix with $0 \leq \boldsymbol{T}_{ij} \leq 1$ for all $i, j \in N$, $\boldsymbol{A}$ is the adjacency matrix, and the operator $\circ$ is the Hadamard product; and $\boldsymbol{D}$ is the weighted degree matrix of the graph for all outgoing edges, such that $\boldsymbol{D}_{ii} = \Delta_i$ and $\boldsymbol{D}_{ij} = 0$ for all $i \neq j$. The matrix resulting from $\boldsymbol{I} - \boldsymbol{D}^{-1}\boldsymbol{W}$ is a normalized Laplacian matrix $\boldsymbol{D}^{-1}\boldsymbol{L} = \boldsymbol{D}^{-1}(\boldsymbol{D} - \boldsymbol{W})$. Note how Equation 6.3 takes the form of the distributed consensus algorithm $\dot{\boldsymbol{x}} = -\boldsymbol{L}\boldsymbol{x}$.

It is well-known that the discrete-time collective dynamics for the consensus problem can also be written as

$$\boldsymbol{x}(k+1) = \boldsymbol{P}\boldsymbol{x}(k) \tag{6.4}$$

where $\boldsymbol{P} = \boldsymbol{I} - \epsilon\boldsymbol{L}$ and $\epsilon > 0$. Here, $\boldsymbol{P}$ is known as the Perron matrix of graph $\mathcal{G}$ with parameter ϵ. If we substitute the normalized Laplacian for $\boldsymbol{L}$ in $\boldsymbol{P}$, then the collective dynamics of the network under our algorithm are

$$\boldsymbol{x}(k+1) = \big((1-\epsilon)\boldsymbol{I} + \epsilon\boldsymbol{D}^{-1}\boldsymbol{W}\big)\boldsymbol{x}(k) \tag{6.5}$$

We now present certain results that are well-known in the consensus literature. These are included mainly for the benefit of the reader to understand how we can claim the asymptotic convergence of protocol 6.3.

Lemma 6.1: Let $\mathcal{G}$ be a digraph with $|N|$ nodes. Then the Perron matrix $\boldsymbol{P}$ with parameter $0 < \epsilon < 1$ is a row-stochastic, non-negative matrix with a trivial eigenvalue of 1.

Proof: Since $\boldsymbol{P} = \boldsymbol{I} - \epsilon \boldsymbol{D}^{-1}\boldsymbol{L}$, we get $\boldsymbol{P}1 = 1 - \epsilon \boldsymbol{D}^{-1}\boldsymbol{L}1 = 1$, which means the row sums of $\boldsymbol{P}$ are 1. Thus, $\boldsymbol{P}$ is row-stochastic and has 1 as a trivial eigenvalue of $\boldsymbol{P}$ for all graphs since zero is an eigenvalue of $\boldsymbol{L}$ associated with the eigenvector 1. Furthermore, we notice that, by definition, the weighted adjacency matrix $\boldsymbol{W}$ is a non-negative matrix. Thus, $\epsilon \boldsymbol{D}^{-1}\boldsymbol{W}$ is also non-negative. Also, $(1 - \epsilon)\boldsymbol{I}$ is always non-negative for $0 < \epsilon < 1$. Since the sum of two non-negative matrices is a non-negative matrix, $\boldsymbol{P}$ is a non-negative matrix. This completes the proof.

Lemma 6.2: Let $\mathcal{G}$ be a digraph with $|N|$ nodes. If $\mathcal{G}$ is strongly connected, then $\boldsymbol{P}$ is a primitive matrix with parameter $0 < \epsilon < 1$.

Proof: An irreducible stochastic matrix is primitive when it has only one eigenvalue with a maximum modulus. Since $\mathcal{G}$ is strongly connected, then $\boldsymbol{P}$ is irreducible [94] and by Lemma 6.1, $\boldsymbol{P}$ is also stochastic. And according to the Perron-Frobenius theorem [132], the fact that $\boldsymbol{P}$ has an eigenvalue of 1 with a positive eigenvector 1 implies that this eigenvalue is the Perron root $\hat{\lambda} = 1$. Hence, any other modulus of eigenvalues of $\boldsymbol{P}$ must be strictly smaller than the Perron root (i.e. $|\lambda| < \hat{\lambda}$ for all eigenvalues λ of $\boldsymbol{P}$, such that $\lambda \neq \hat{\lambda}$). Thus, $\boldsymbol{P}$ is a primitive matrix.

Theorem 6.1 (Convergence): Consider a network of agents $\boldsymbol{x}_i(k+1) = \boldsymbol{x}_i(k) + \epsilon \boldsymbol{u}_i(k)$ for a fixed, strongly connected graph $\mathcal{G}$ that applies the distributed consensus protocol 6.3. Then, protocol 6.3 asymptotically solves a consensus problem.

Proof: Considering that $\boldsymbol{x}(k) = \boldsymbol{P}^k \boldsymbol{x}(0)$, consensus is reached in discrete-time if $\lim_{k\to\infty} \boldsymbol{P}^k$ exists. According to the Perron-Frobenius theorem [132], this limit exists for primitive matrices and according to Lemma 6.2, $\boldsymbol{P}$ is a primitive matrix under the conditions of protocol 6.3. Thus, protocol 6.3 asymptotically solves the consensus problem.

6.3 Trust-Based Consensus with Static Trust

In this section, we study the trust-based consensus algorithm under the condition of static trust for a specific case. We utilize a simple three-agent directed network (Figure 6.1) with the following adjacency matrix.

$$\boldsymbol{A} = \begin{bmatrix} 1 & 0 & 1 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \end{bmatrix} \tag{6.6}$$

This network structure is loosely constructed with the convoy application in mind, where each agent in the convoy observes and attempts to converge to agreement

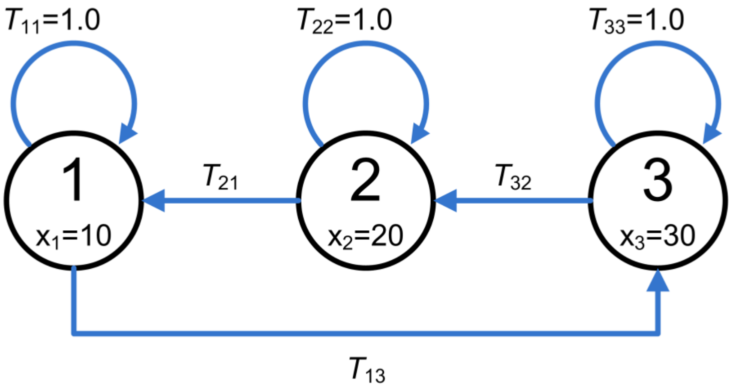


Figure 6.1. Three-Agent Strongly-Connected Network for the Trust-Based Consensus Study.

with its local leader. Note, however, that the global leader (agent 1) does not have a natural local leader. As such, in order to ensure a strongly-connected graph (as required by Theorem 6.1), the global leader is required to observe and attempt to converge to agreement with the last agent in the convoy.

Our intent with this study is to establish convergence value and time step results for the trust-based consensus protocol through simulation, and understand how trust impacts these results. For each simulation, we initialize $N = \{1,2,3\}$, $\boldsymbol{x}(0) = [10,20,30]^{\mathrm{T}}$ and $\epsilon = 0.1$; and terminate when $\sum_{i,j \in N} \left\| \boldsymbol{x}_i - \boldsymbol{x}_j \right\| < 1^{-5}$. Each simulation runs the trust-based consensus protocol in Equation 6.3 using a trust matrix with the constraints in 6.7 to the precision of 10^{-2}.

$$\boldsymbol{T} = \begin{bmatrix} 1 & 0 & \boldsymbol{T}_{13} \\ \boldsymbol{T}_{21} & 1 & 0 \\ 0 & \boldsymbol{T}_{32} & 1 \end{bmatrix} \qquad \begin{array}{c} 0 \leq \boldsymbol{T}_{13} \leq 1 \\ 0.01 \leq \boldsymbol{T}_{21} \leq 1 \\ 0.01 \leq \boldsymbol{T}_{32} \leq 1 \end{array} \tag{6.7}$$

We show the results of an example run of one such simulation in Figure 6.2, with $\boldsymbol{T}_{13} = 0.2$, $\boldsymbol{T}_{21} = 0.3$, and $\boldsymbol{T}_{32} = 0.4$.

Our extensive study applied Equation 6.3 to every possible combination of a trust matrix, subject to the constraints in Equation 6.7, which resulted in a total of 1.01 million trust-based consensus simulations. The results for each simulation consisted of a 5-dimensional vector: three dimensions for the trust values of each agent, one dimension for the final consensus value, and one dimension for the number of time steps to reach the consensus value. Using the entire collection of these vectors, we generated data visualizations to gain an understanding of how changes in trust influence the consensus values and time steps.

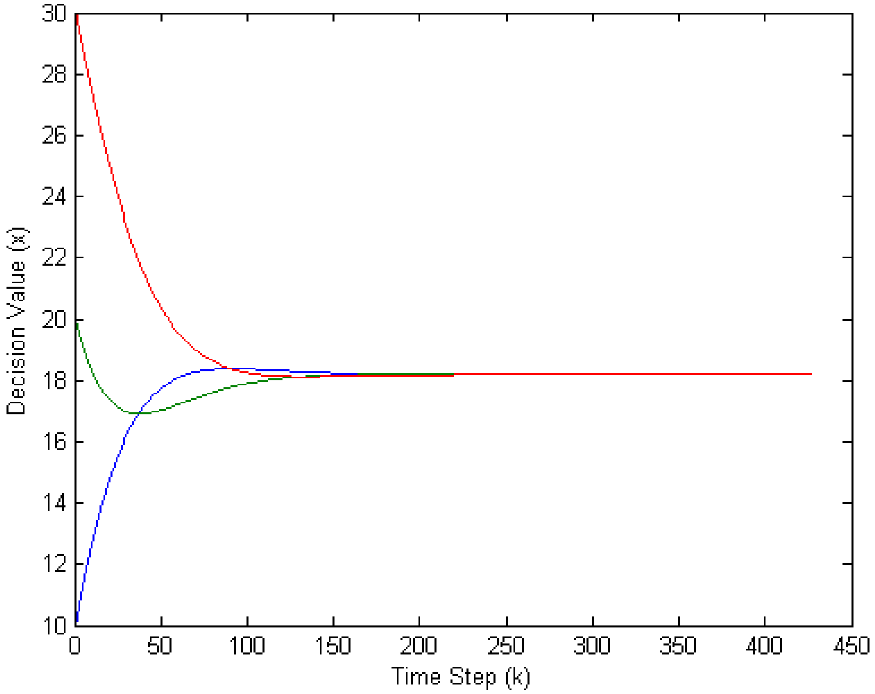
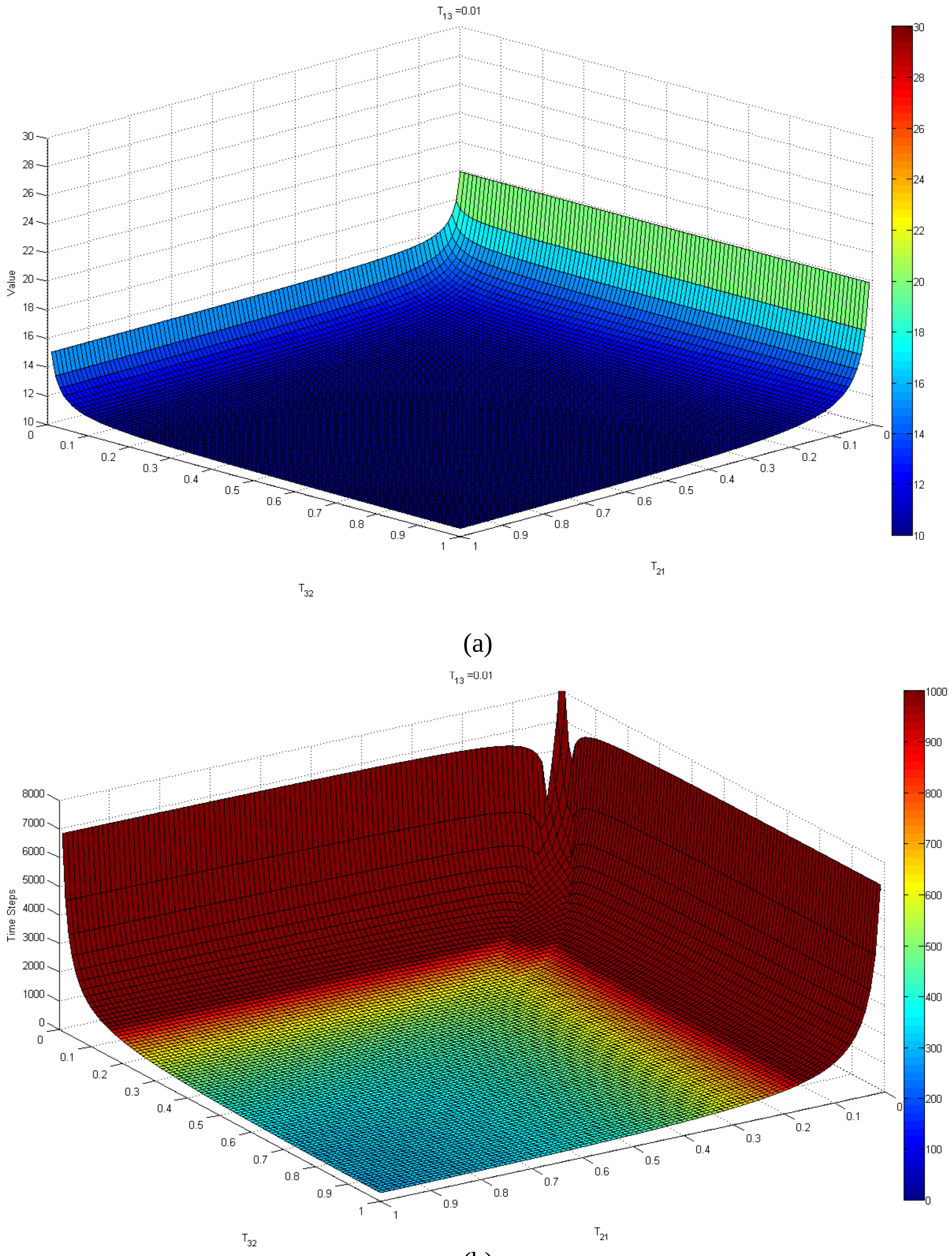


Figure 6.2. Example Run of the Trust-Based Consensus Protocol with $\boldsymbol{T}_{13} = 0.2$, $\boldsymbol{T}_{21} = 0.3$, and $\boldsymbol{T}_{32} = 0.4$. A consensus agreement is reached at value 18.1928 after 427 time steps.

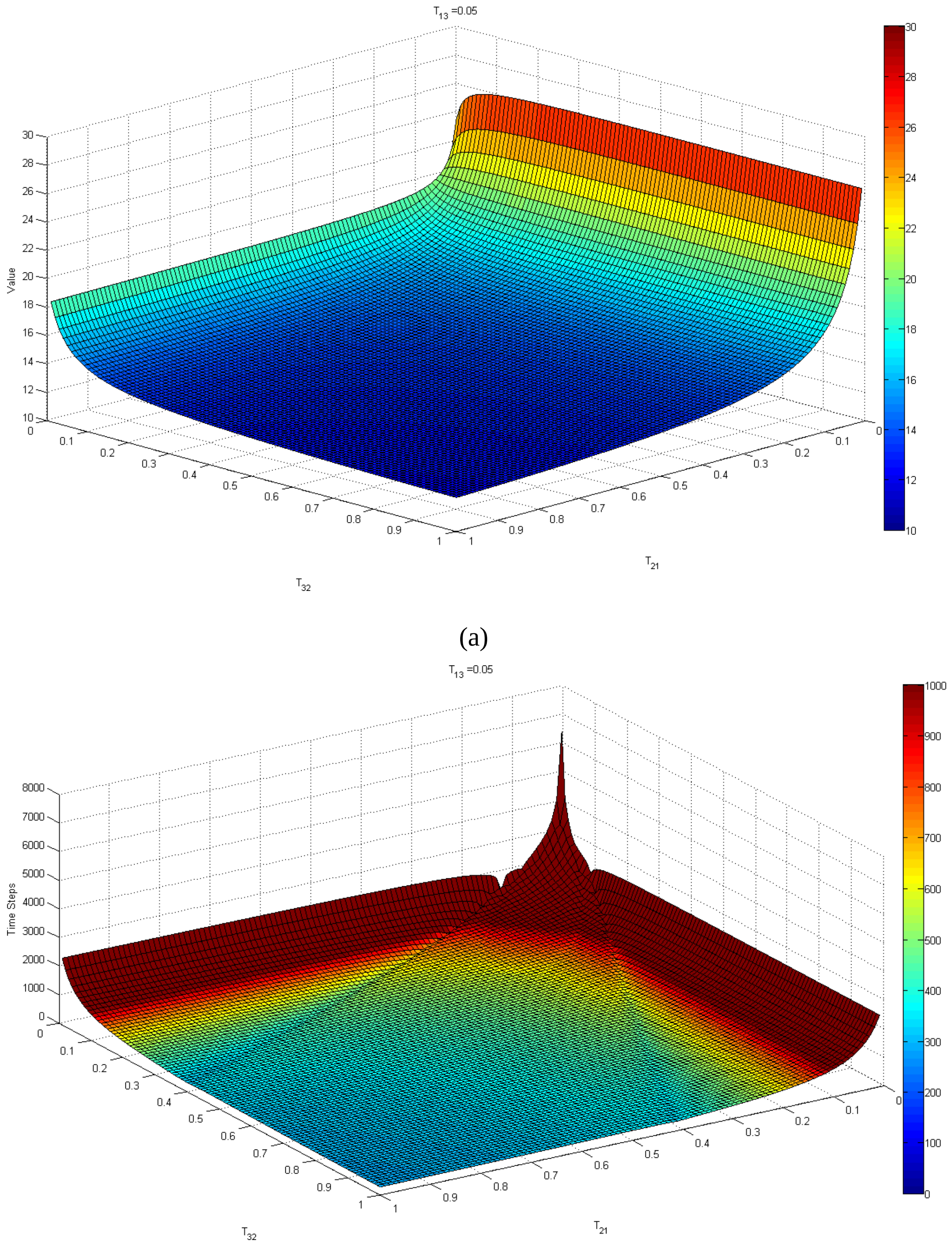
In our first set of visualizations, represented in Figures 6.3 through 6.8, we held the trust value $\boldsymbol{T}_{13}$ fixed and plotted the consensus value (a) and time steps (b) with respect to changes in $\boldsymbol{T}_{21}$ and $\boldsymbol{T}_{32}$. In Figure 6.3, $\boldsymbol{T}_{13} = 0.01$. In this figure, the value surface plot shows the consensus value generally settling near the original value of agent 1, namely 10. This is because agent 1 has extremely low trust toward agent 3, and therefore, converges much slower towards the decision value of agent 3. The effect of extremely low trust is clearly depicted in the time step surface plot, where the number of time steps necessary for convergence easily exceeds 1000 when any two



(a)

(b)

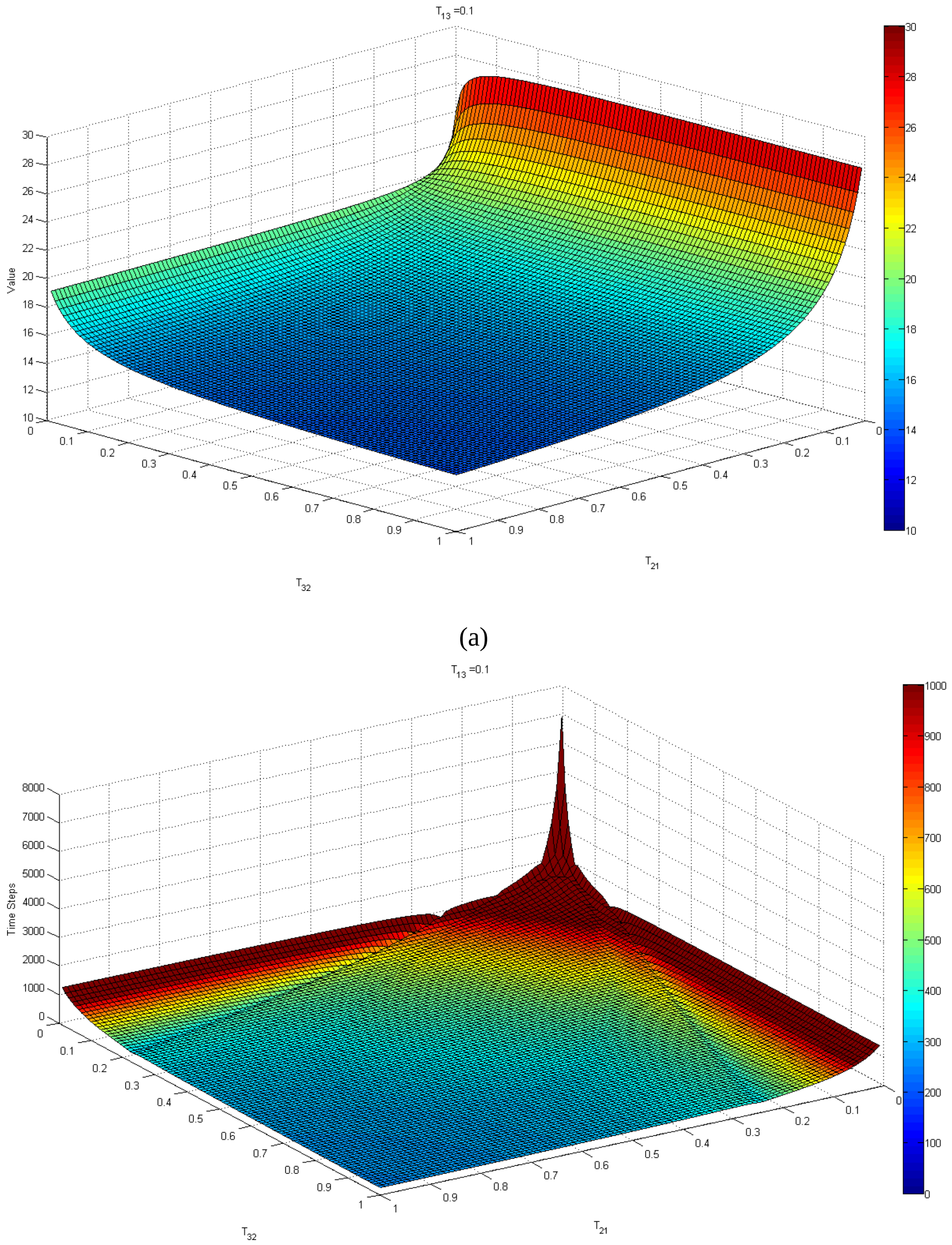
Figure 6.3. Value (a) and Time Step (b) Convergence Results for $\boldsymbol{T}_{13} = 0.01$, $0.01 \leq \boldsymbol{T}_{21} \leq 1$, and $0.01 \leq \boldsymbol{T}_{32} \leq 1$. (This figure is presented in color; the black and white reproduction may not be an accurate representation.)



(a)

(b)

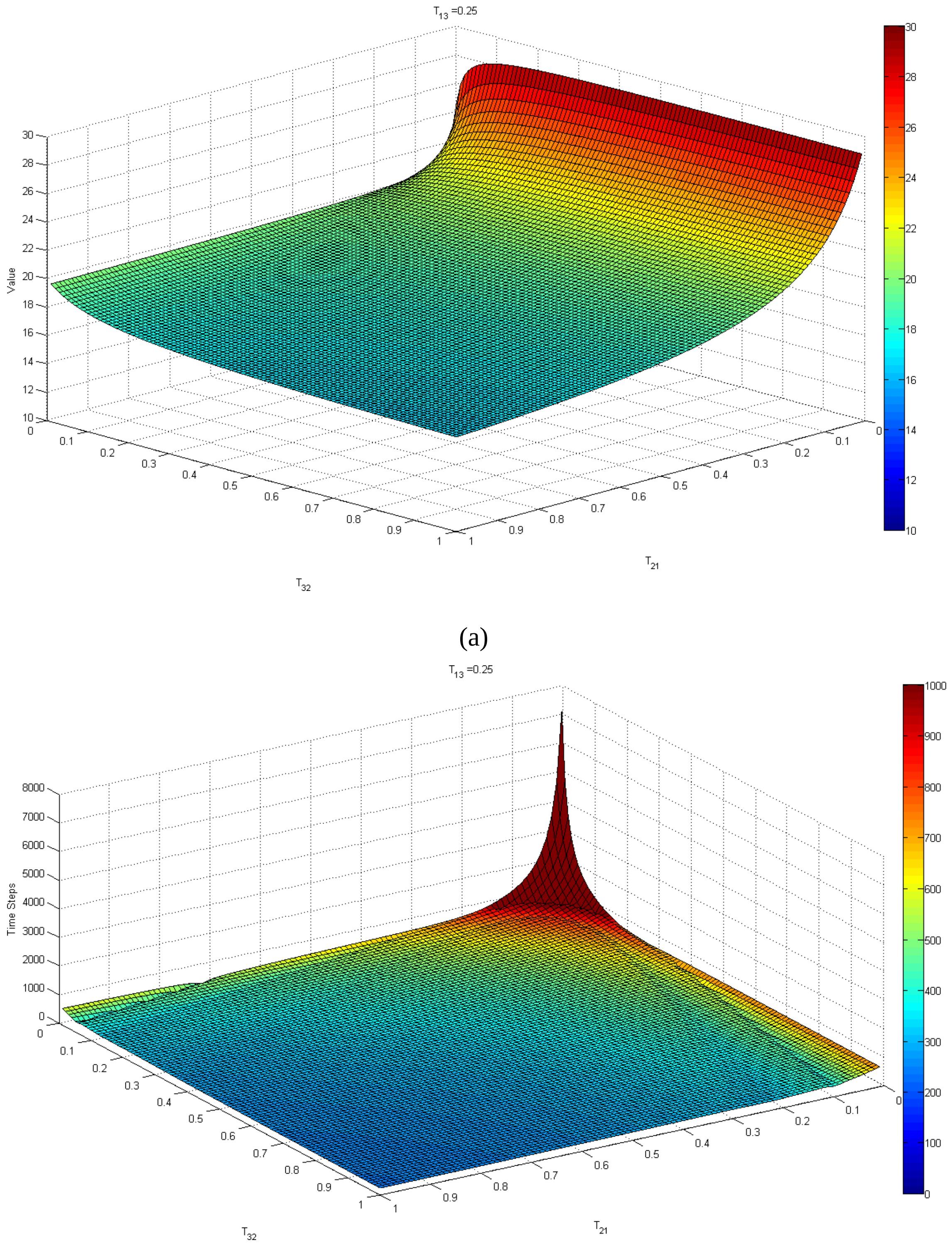
Figure 6.4. Value (a) and Time Step (b) Convergence Results for $\boldsymbol{T}_{13} = 0.05$, $0.01 \leq \boldsymbol{T}_{21} \leq 1$, and $0.01 \leq \boldsymbol{T}_{32} \leq 1$. (This figure is presented in color; the black and white reproduction may not be an accurate representation.)



(a)

(b)

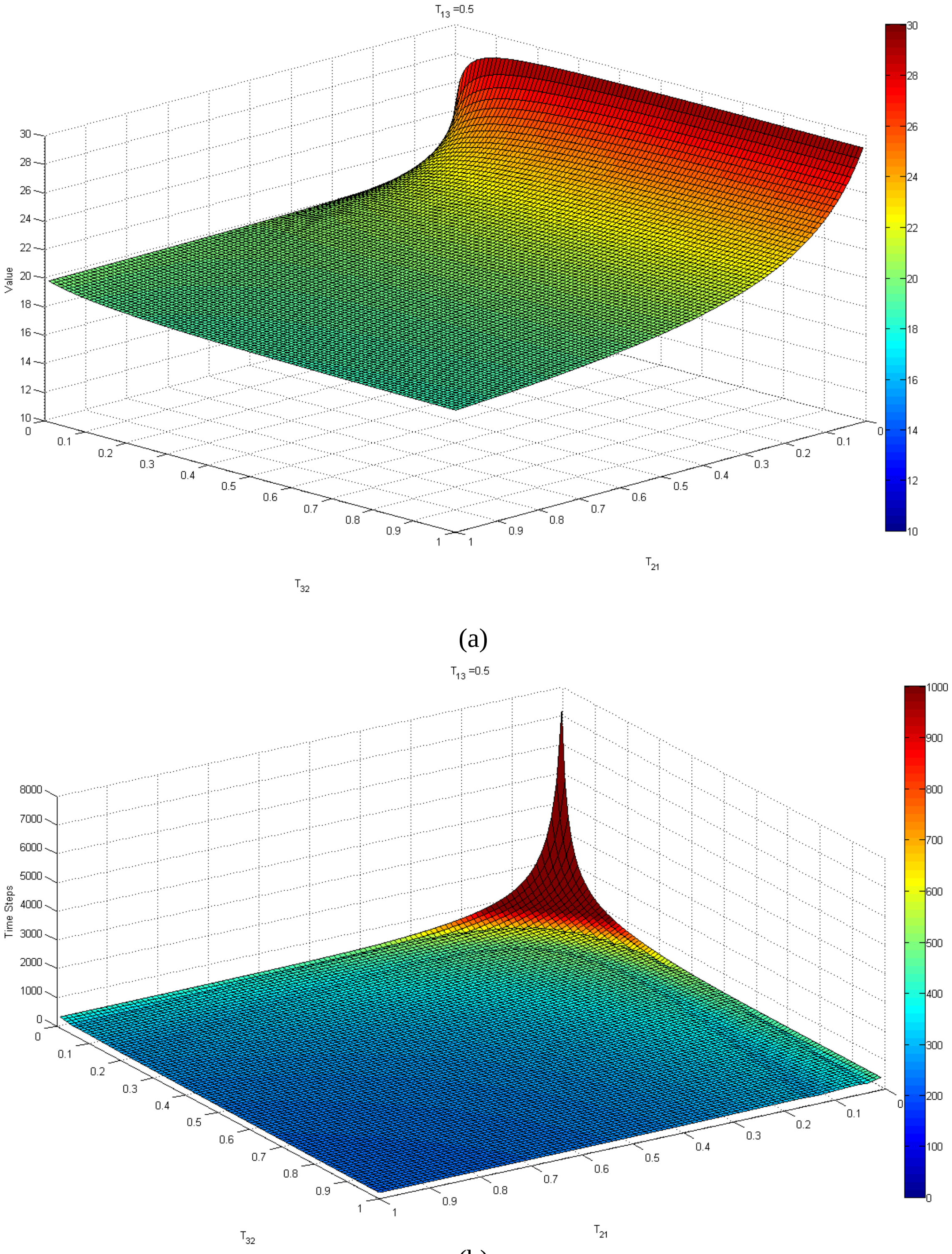
Figure 6.5. Value (a) and Time Step (b) Convergence Results for $\boldsymbol{T}_{13} = 0.1$, $0.01 \leq \boldsymbol{T}_{21} \leq 1$, and $0.01 \leq \boldsymbol{T}_{32} \leq 1$. (This figure is presented in color; the black and white reproduction may not be an accurate representation.)



(a)

(b)

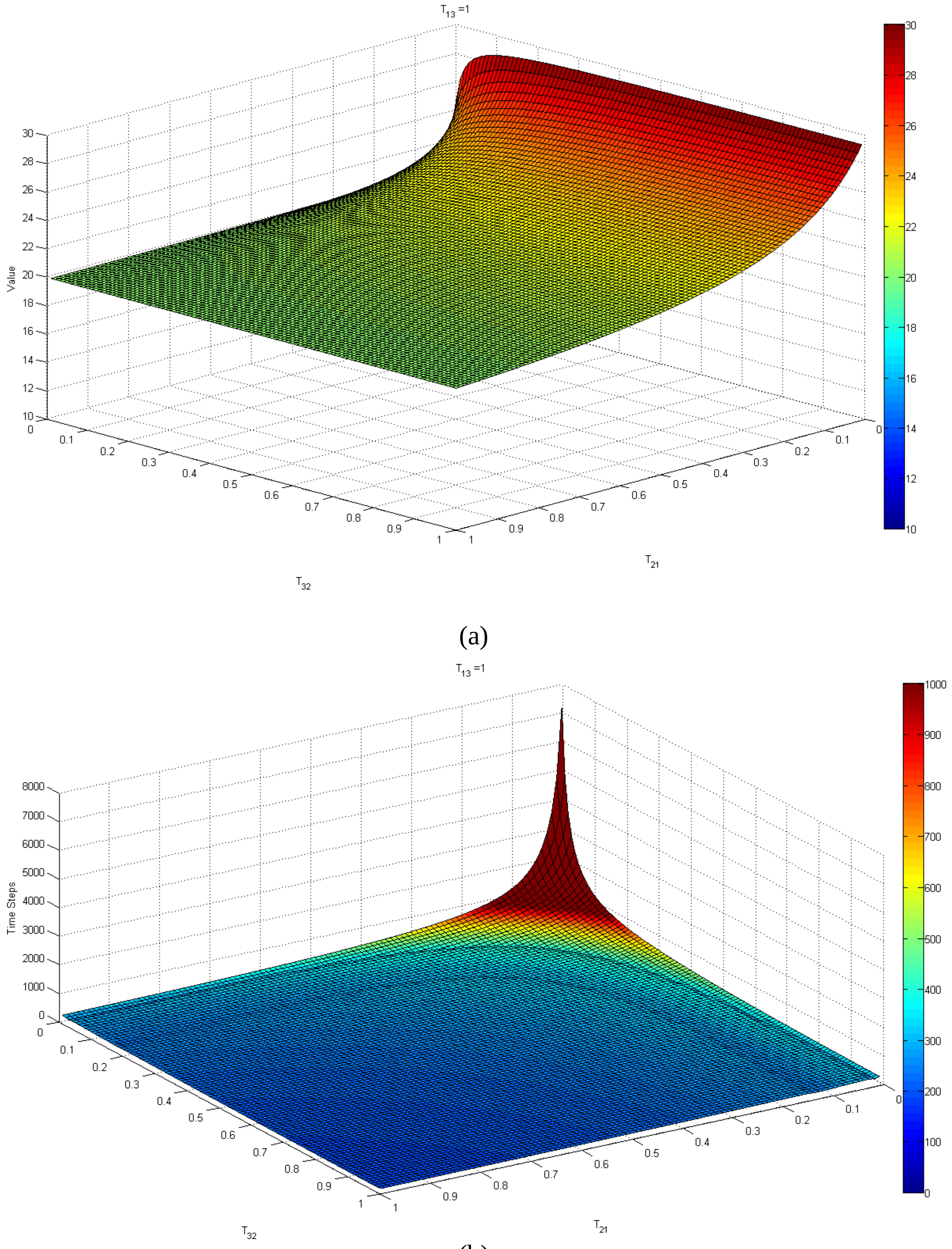
Figure 6.6. Value (a) and Time Step (b) Convergence Results for $\boldsymbol{T}_{13} = 0.25$, $0.01 \leq \boldsymbol{T}_{21} \leq 1$, and $0.01 \leq \boldsymbol{T}_{32} \leq 1$. (This figure is presented in color; the black and white reproduction may not be an accurate representation.)



(a)

(b)

Figure 6.7. Value (a) and Time Step (b) Convergence Results for $\boldsymbol{T}_{13} = 0.5$, $0.01 \leq \boldsymbol{T}_{21} \leq 1$, and $0.01 \leq \boldsymbol{T}_{32} \leq 1$. (This figure is presented in color; the black and white reproduction may not be an accurate representation.)



(a)

(b)

Figure 6.8. Value (a) and Time Step (b) Convergence Results for $\boldsymbol{T}_{13} = 1$, $0.01 \leq \boldsymbol{T}_{21} \leq 1$, and $0.01 \leq \boldsymbol{T}_{32} \leq 1$. (This figure is presented in color; the black and white reproduction may not be an accurate representation.)

agents have low trust; and dramatically exceeds 8000 when all three agents have low trust for each other.

In Figure 6.4, $\boldsymbol{T}_{13}$ is increased to 0.05 and the effects of this small change are markedly noticeable. With regards to the value surface plot, we see the consensus values rise, noting the presence of more cyan through red coloring on the surface. On the time step surface plot, the number of time steps along the trust value axes have dramatically reduced, from nearly 7000 in Figure 6.3b to just slightly over 2000. One prominent feature on the time step surface, which will remain a permanent fixture in all of the remaining time step surface plots, is the time step spike near the lowest trust values of $\boldsymbol{T}_{21}$ and $\boldsymbol{T}_{32}$. This spike indicates that extremely low trust for any two agents in this system negates the effects of any level of trust of the third agent in practically increasing convergence speed.

In Figure 6.5, we increase $\boldsymbol{T}_{13}$ to 0.1. The value surface plot, like before, exhibits a general rise in consensus values. This trend will continue through Figures 6.6, 6.7, and 6.8, gradually eliminating all shades of blue and cyan in the value surface plots. We reinforce this assertion with Figure 6.9, which shows monotonically increasing statistics about the final consensus value in relation to the trust value. The time step surface plots in Figure 6.5 through 6.8, also like before, show a general decrease in overall number of time steps necessary to achieve consensus. This indicates a direct correlation between the level of trust in a network and convergence speed – that is, as trust levels generally increase in a network, the convergence speed of the consensus protocol also increases. We reinforce this assertion with Figure 6.10, which

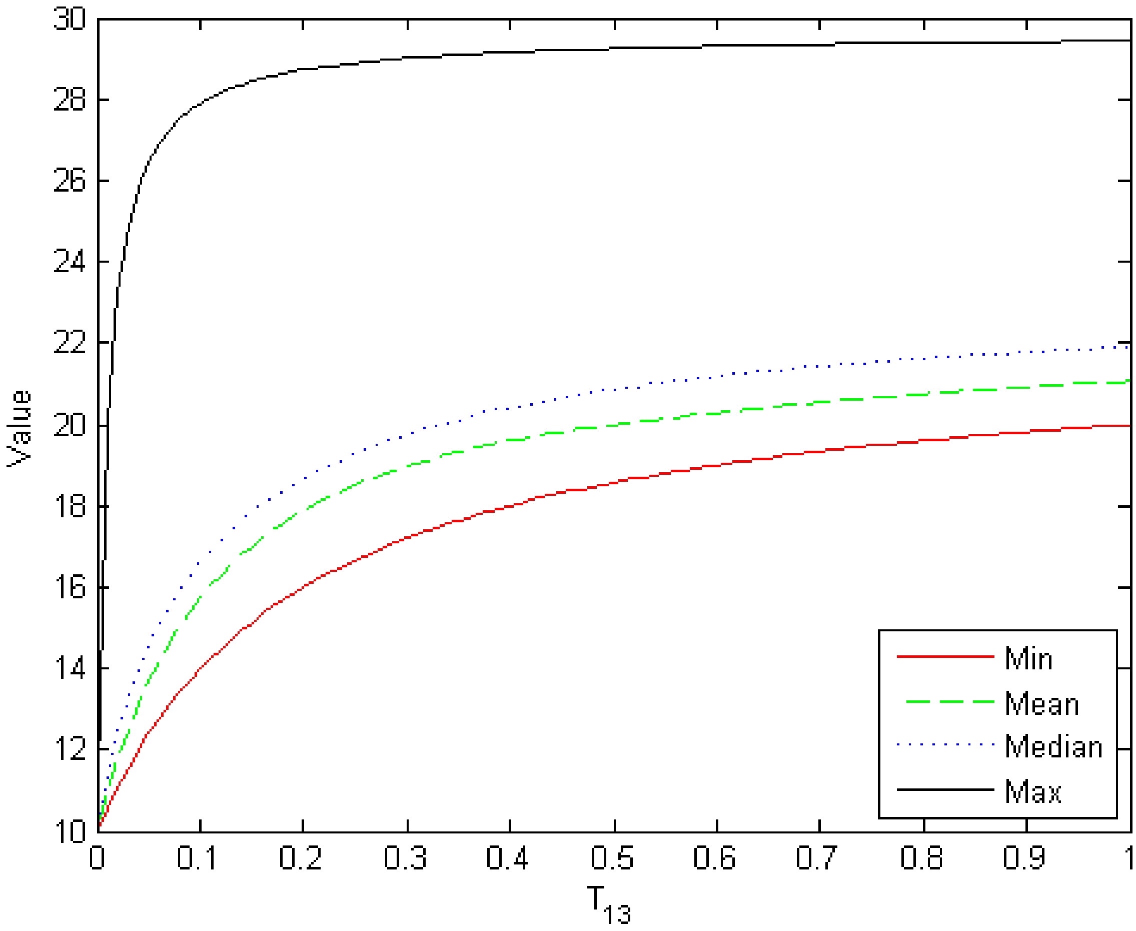


Figure 6.9. Statistics about the Final Consensus Value in Relation to the Trust Value (This figure is presented in color; the black and white reproduction may not be an accurate representation.)

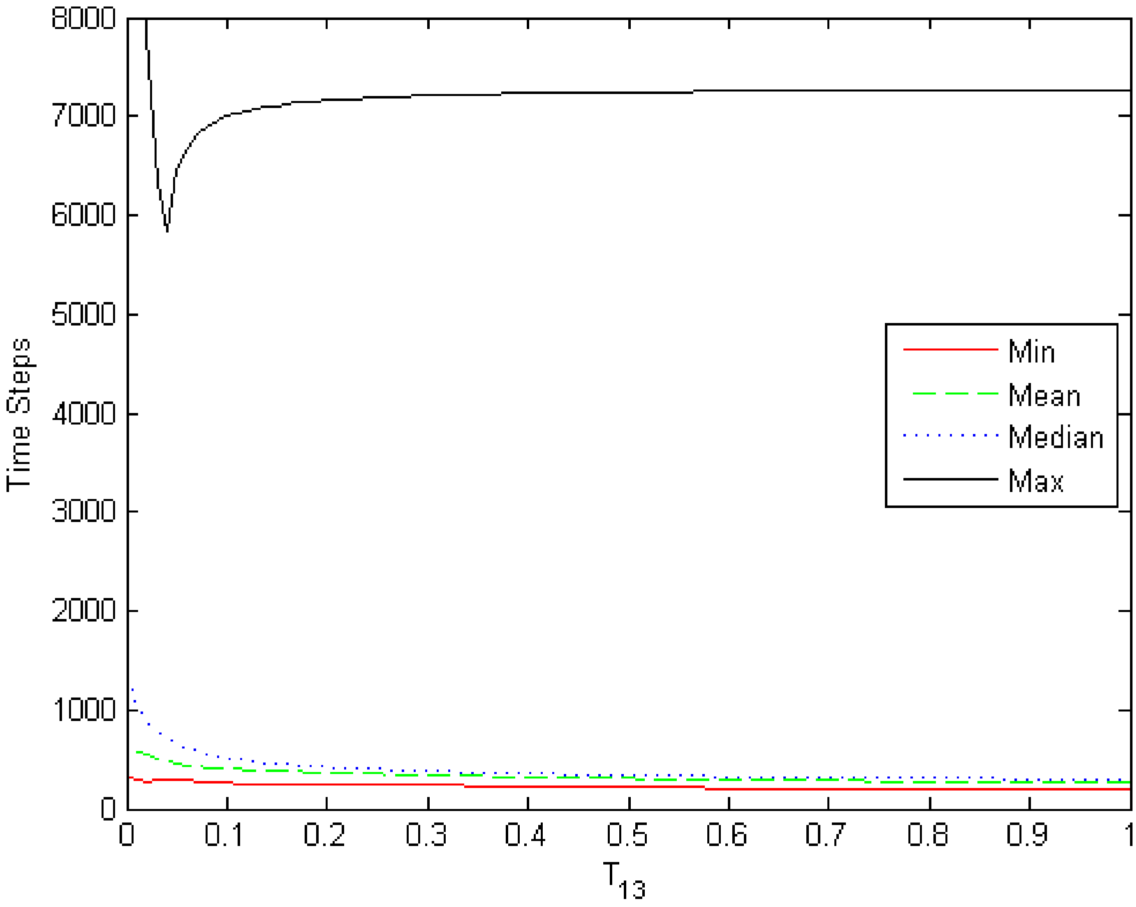


Figure 6.10. Statistics about the Final Consensus Time Steps in Relation to the Trust Value. (This figure is presented in color; the black and white reproduction may not be an accurate representation.)

shows monotonically decreasing statistics (in min, mean, and median) for time steps in relation to the trust value.

In general, a trust consensus protocol with static trust assumes that the trust values are known prior to the execution of the consensus protocol. This assumption may be particularly useful in sensor fusion applications of similar sensors, where each sensor may be known to degrade in its calibration at a known rate. Trust values can be extrapolated from calibration degradation curves using an application-specific formula.

6.4 Trust-Based Consensus with Dynamic Trust

In this section, we study the trust-based consensus algorithm under the condition of dynamic trust for a specific case. We utilize the same three-agent directed network (Figure 6.1) used in Section 6.3 with the adjacency matrix given in Equation 6.6.

The use of dynamic trust updates during the consensus process signify that agents are interested in cultivating trust with other agents with respect to particular contexts during the consensus process. Thus, unlike in the static-trust case, agents do not need to assume any pre-determined trust values. Rather, trust values are directly dependent on the agent behaviors during the consensus process. For our study, we use the RoboTrust algorithm in Equation 5.10 to generate the trust updates during the consensus process.

In order to use the RoboTrust algorithm, we must first define at least one acceptance function, which describes a particular context of acceptable regions in a feature space. Our choice of an acceptance function for our purposes is arbitrary; normally, acceptance functions are designed with a specific application in mind. Thus, we consider the context of an agent's "willingness to cooperate" to reach an agreement during consensus. We measure an agent's willingness to cooperate by evaluating whether a particular action demonstrates a willingness to shorten the distance δ between disagreements. More specifically, we say that agent i observes agent j favorably if agent j's current state vector is closer to agent i's previous state vector than agent j's previous state vector.

$$\delta_{ij} = \left\|\boldsymbol{x}_i(k-1) - \boldsymbol{x}_j(k-1)\right\| - \left\|\boldsymbol{x}_i(k-1) - \boldsymbol{x}_j(k)\right\| \tag{6.8}$$

$$i \neq j, k > 0$$

$$\boldsymbol{z}_{ij}(k) = \begin{cases} 1 & \delta_{ij} > 0 \\ 0 & \delta_{ij} < 0 \end{cases} \tag{6.9}$$

In the event that $\delta_{ij} = 0$ (i.e. $\boldsymbol{x}_j(k-1) = \boldsymbol{x}_j(k)$), there is some ambiguity since it is not clear whether or not agent j intends to cooperate with agent i. Agent j may have chosen to not change for different reasons, not all of which may indicate malicious intent or unwillingness to cooperate. For example, agent j may have become isolated from its first neighbors and has no way to converge to any consensus value. Note that agent i does not need to be a first-neighbor of agent j, even if agent i is able to know agent j's decision value, since it may be the case that $\boldsymbol{A}_{ji} = 0$ even if $\boldsymbol{T}_{ji} > 0$.

Handling this ambiguity is somewhat arbitrary and may be dependent on the specific application. For example, in a real application, a practitioner may choose to resolve the ambiguity by making the result equal to a binary random variable taken from some probability distribution. However, since we intend to study the consensus protocol in a deterministic manner, we simply assign the acceptance value as the result of a negation on the previous acceptance value in the event there is no change.

$$\boldsymbol{z}_{ij}(k) = \{0,1\} \backslash \boldsymbol{z}_{ij}(k-1) \qquad \delta_{ij} = 0 \tag{6.10}$$

With the acceptance function established, we now focus our study to understand how the tolerance and confirmation parameters, τ and c, in the RoboTrust algorithm influence the trust-based consensus algorithm with respect to its final consensus value and the number of time steps to reach consensus. As before, we initialize $N = \{1,2,3\}$, $\boldsymbol{x}(0) = [10,20,30]^{\mathrm{T}}$, and $\epsilon = 0.1$; and terminate when $\sum_{i,j \in N} \left\| \boldsymbol{x}_i - \boldsymbol{x}_j \right\| < 1^{-5}$. In addition, we set the initial trust matrix equal to the identity matrix (i.e. $\boldsymbol{T} = \boldsymbol{I}$). The rationale behind this trust initialization is based on the assumption that no agent enters the consensus with any previous trust-based opinions, and thus, will need to cultivate trust during the consensus process.

We show the results of an example run of the trust-based consensus protocol with dynamic trust in Figure 6.11. A visual comparison between Figure 6.2 and Figure 6.11 shows drastically different time series for the decision values. In particular, we see time series sections in Figure 6.11 where $\boldsymbol{u}_i(k) = 0$ “before” an agent reaches the final convergence result. These sections represent portions of the convergence process when a particular agent i becomes isolated from its first-neighbors due to a total absence of trust in its first-neighbors. For example, at $k = 42$, the decision values of agent 1 and 3 intersect. At the next time step, agent 1 begins to lose some trust toward agent 3 because agent 3 appears to be increasing the disagreement distance. Eventually, by $k = 46$, agent 1 loses all trust for agent 3 (its only first-neighbor) and flatlines till $k = 68$, which interestingly is an intersection point between agent 2 and agent 3. Figure 6.12 shows a detailed representation of the Figure 6.11 to illustrate this clearer.

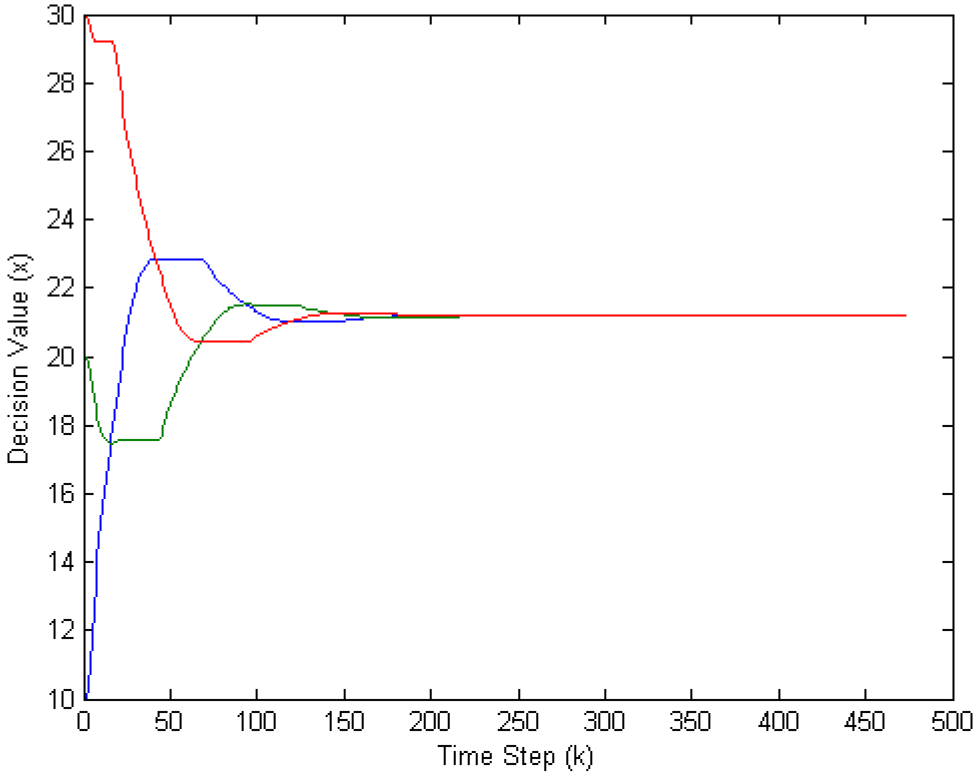


Figure 6.11. Example Run of the Trust-Based Consensus with RoboTrust. A consensus agreement is reached at value 21.1814 after 473 time steps with each agent using the RoboTrust algorithm with parameters τ=3 and $c = 5$. (This figure is presented in color; the black and white reproduction may not be an accurate representation.)

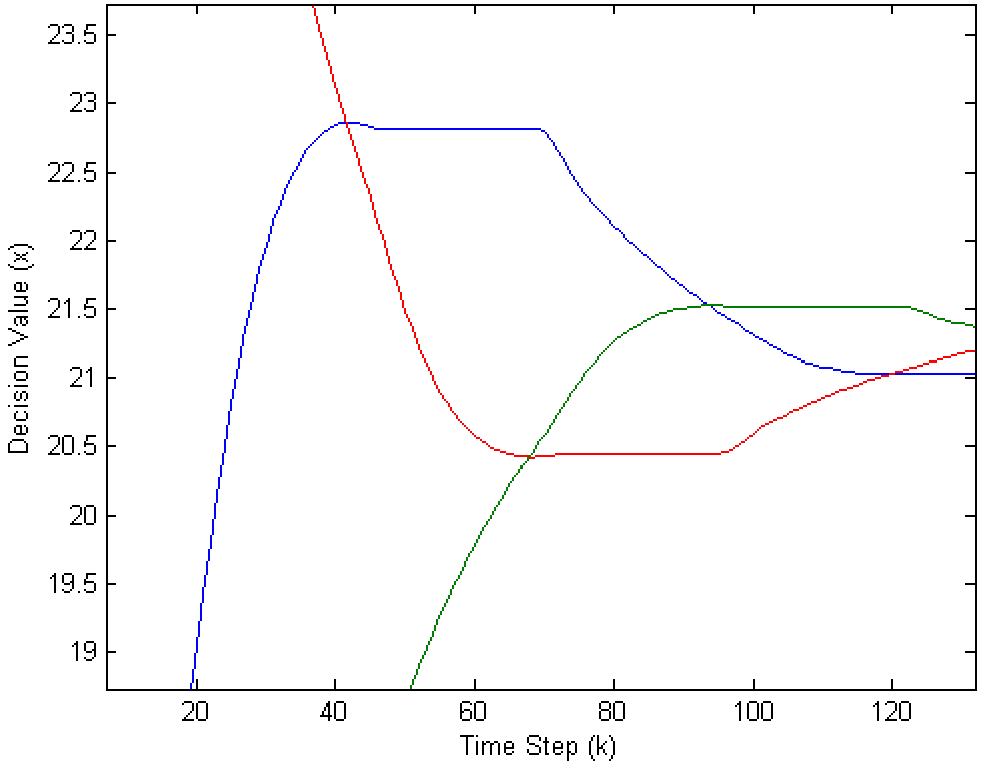


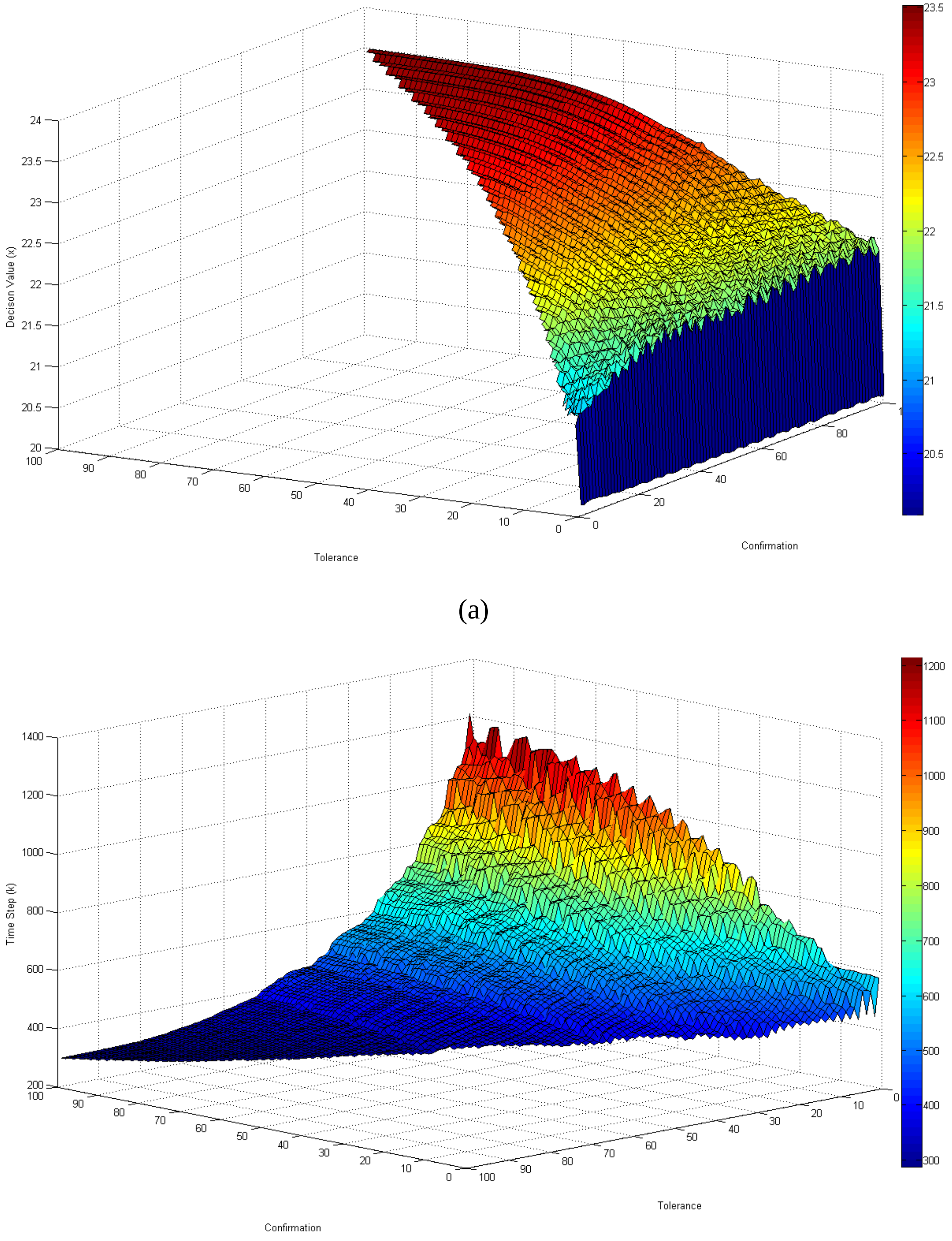
Figure 6.12. A Closer View of the Time Series in Figure 6.11. Agents 1 (blue) and 3 (red) intersect at $k = 42$. Agents 2 (green) and 3 (red) intersect at $k = 68$. (This figure is presented in color; the black and white reproduction may not be an accurate representation.)

6.4.1 Dynamic Trust with Same Tolerance and Confirmation Parameters

For our first study, we executed the trust-based consensus protocol with RoboTrust updates, initializing the same (τ, c) pair for all agents for each simulation. Our study iterated through each (τ, c) pair within the ranges of $0 \leq \tau \leq 100$ and $1 \leq c \leq 100$ for all agents. This case is important to consider since a practioner with similar (or exactly the same) agents may choose to set the same (τ, c) values in all agents for a particular context for practicality and simplicity reasons. Our results are visually depicted in Figure 6.13. Figure 6.13a shows the value surface plot appearing to reach an asymptotic limit between 23.5 and 24 as tolerance and confirmation values both increase. In Figure 6.13b, we see that higher tolerance values tend to shorten the length of time necessary to reach convergence. Also, higher confirmation values tend to extend the length of time necessary to reach convergence. It is important to note, however, that the minimum time occurs at $(\tau, c) = (85{,}85)$. Thus, one cannot expect that an increase in tolerance and confirmation values is guaranteed to always yield a small number of time steps.

6.4.2 Dynamic Trust with Different Tolerance and Confirmation Parameters

For our second study, we sought to understand the consensus process with dynamic trust with different (τ, c) pairs for each agent. Hence, we executed seven sets of 10,000 trust-based consensus simulations with RoboTrust updates using different randomly-selected (τ, c) pairs for all agents. Random pair assignments were taken from a uniform distribution and constrained by a maximum confirmation parameter c_{max} in order to gauge the effect of the (τ, c) pair diversity in an agent population. In



(a)

(b)

Figure 6.13. Surface Plot of the Consensus Value (a) and Time Steps Needed for Convergence (b) on a (τ, c) Pair. The range represented in these plots are $0 \leq \tau \leq 100$ and $1 \leq c \leq 100$. (This figure is presented in color; the black and white reproduction may not be an accurate representation.)

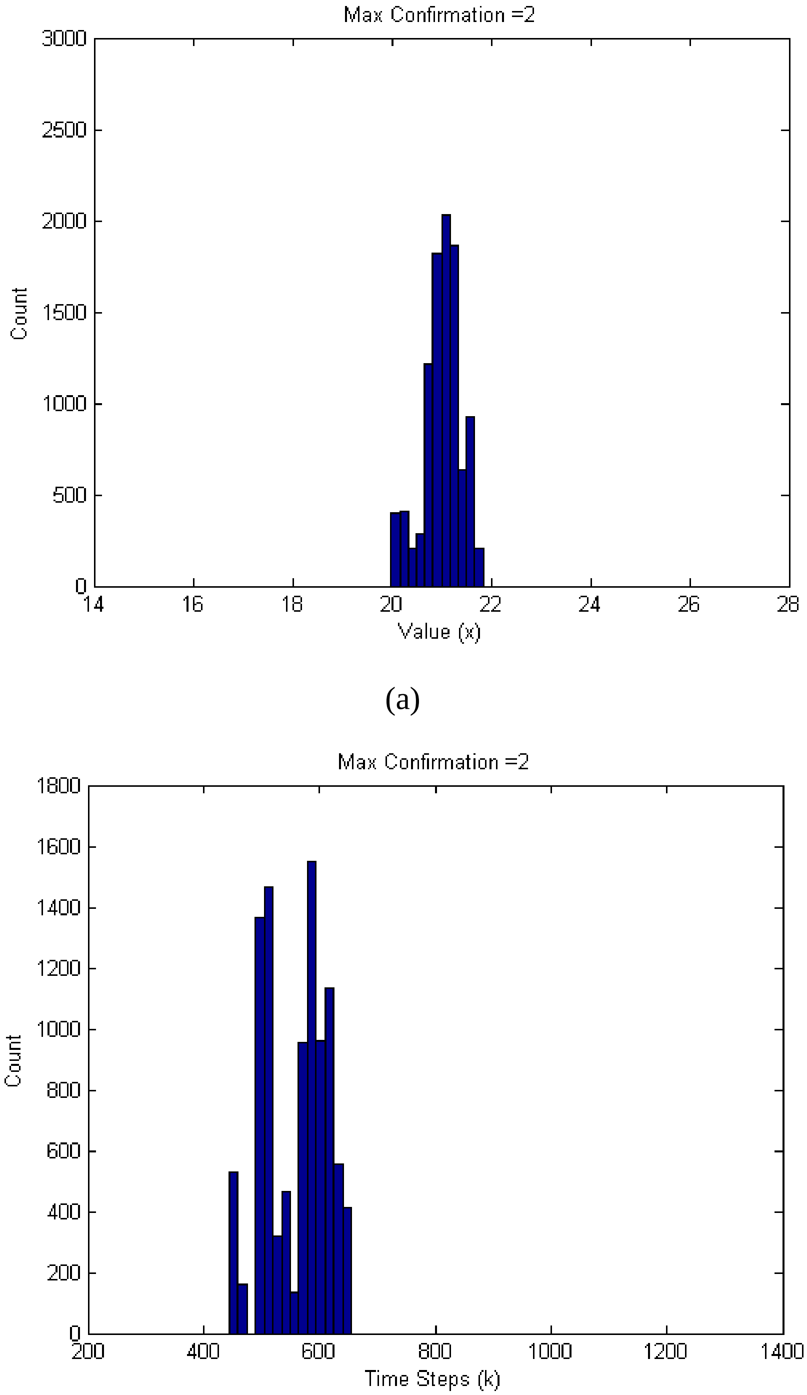
this study, $c_{max} \in \{2, 5, 10, 25, 50, 100, 200\}$. Thus, the random selection of a (τ, c) pair came from a population constrained by $0 \leq c \leq c_{max}$ and $0 \leq \tau \leq c$.

Our results for this study are visually depicted in histograms of the final consensus values and the final number of time steps in Figure 6.14 through Figure 6.20, where each histogram is divided into 60 bins of equal width. The histograms clearly show that an increase in (τ, c) pair diversity in an agent population produces more variety in final consensus results for both the final value and the number of time steps. Statistics taken from the histogram results, shown in Figures 6.21 and 6.22 and Tables 6.1 and 6.2, also support this conclusion with the wide gap between the minimum and maximum values. In addition, these statistics show that an increase in (τ, c) pair diversity generally shortens the time necessary to reach consensus. This is most likely due to the increased likelihood of selecting more tolerant agents, who are able to maintain a higher level of trust for longer periods of time than less tolerant agents for the same observations.

Conclusion

In summary, we provided a distributed, discrete-time, trust-based consensus protocol, mathematically proved its asymptotic convergence, and analyzed it under the cases of static-trust and dynamic-trust in a simple three-agent network. Our static-trust experiment indicated an inverse correlation between the overall level of trust in a network and convergence time – that is, higher trust levels in the network generally decreased the amount of time necessary to reach consensus. Our dynamic-trust experiments used the RoboTrust algorithm to perform trust updates during the consensus process according to each agent’s willingness to cooperate. When all agents were set to have the same confirmation and tolerance parameters in an experiment, we saw that higher tolerance values tended to shorten convergence time while higher confirmation values tended to extend convergence time. When all agents were randomly assigned different confirmation and tolerance parameters in an experiment, we showed that an increase in parameter diversity produced more variety in final consensus results for both the final value and the convergence time.

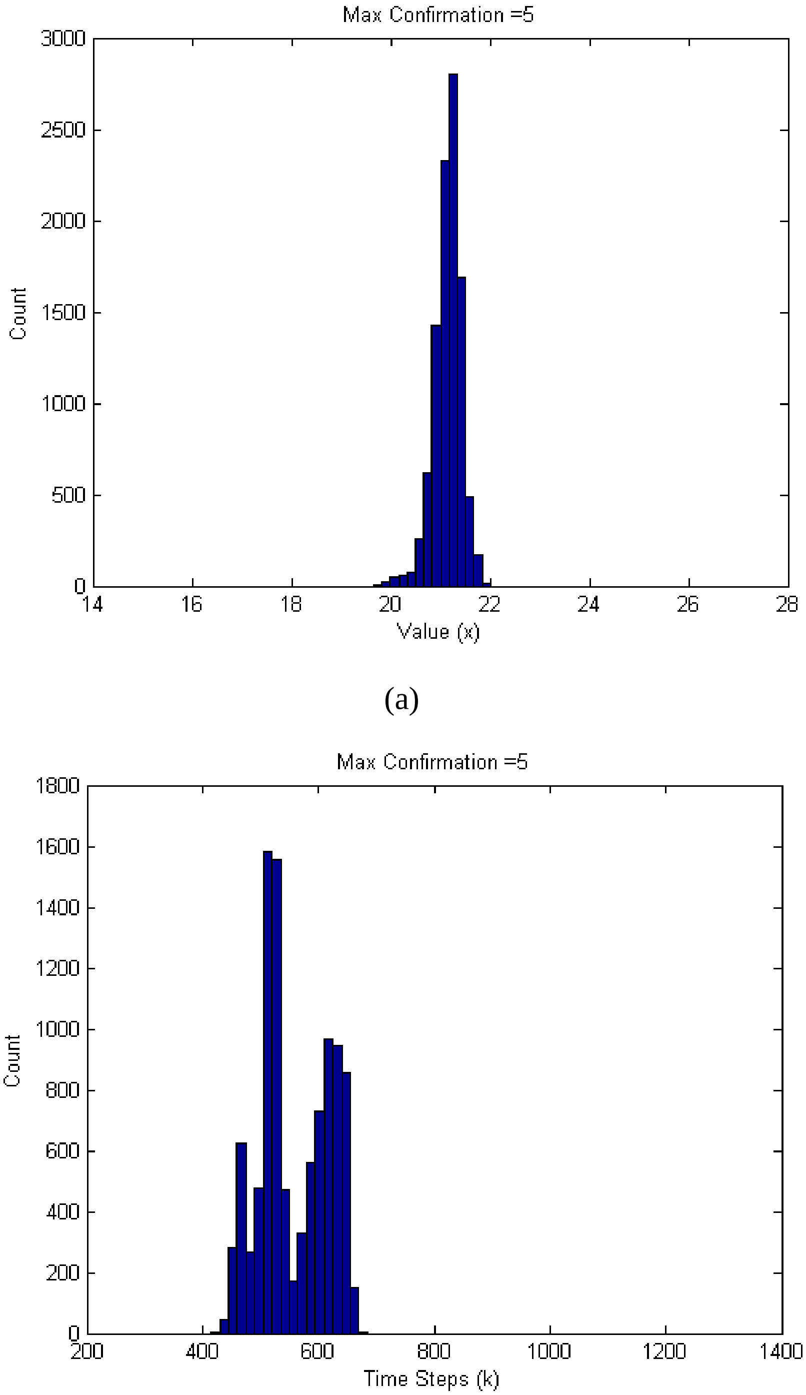
It is important to note that all of our experiments considered only one specific network with specific initial conditions for a specific context. Therefore, the trends witnessed in these studies should not be universally applied to other networks and configurations without proper verification. These experiments serve only to provide baseline conclusions for comparison in future experiments using different network configurations, initial conditions, or contexts.



(a)

(b)

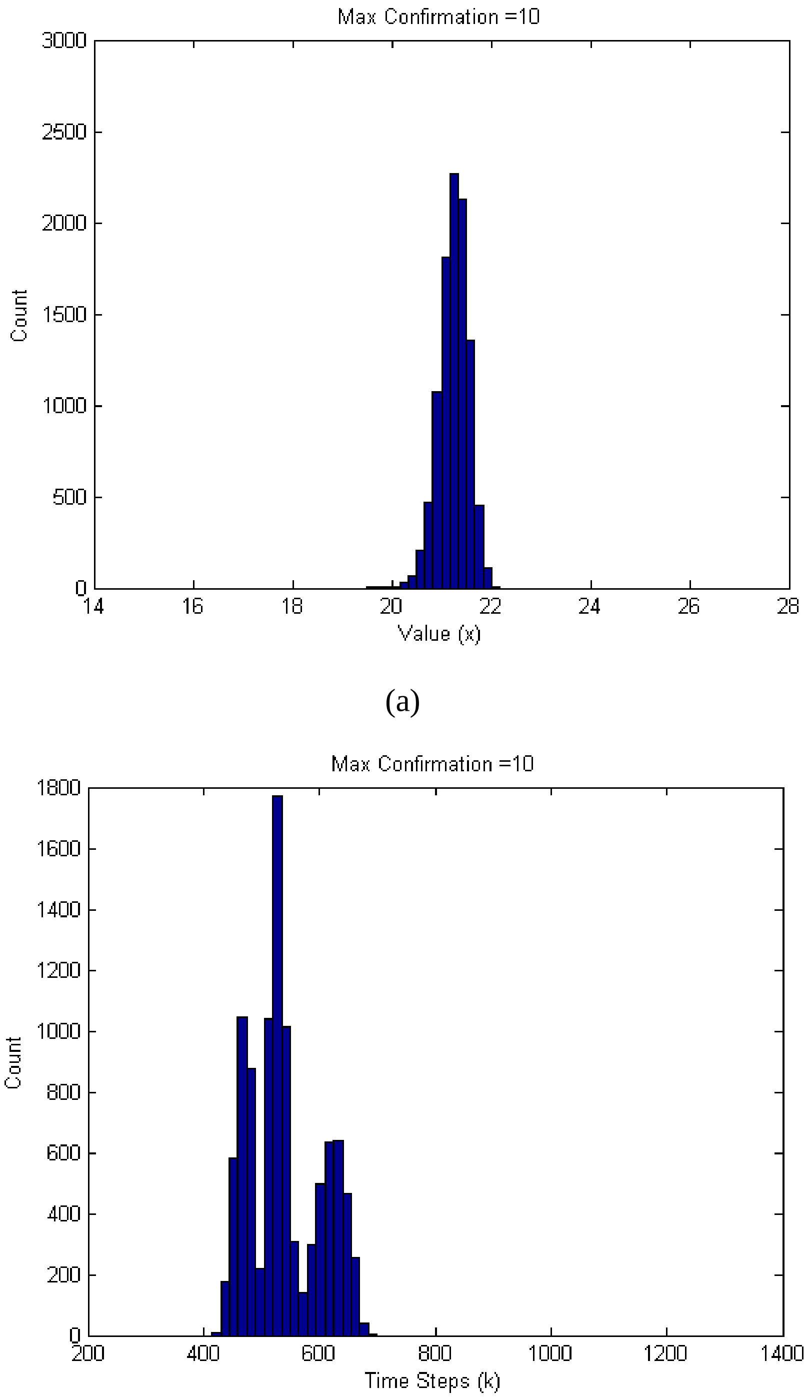
Figure 6.14. Histograms of the Consensus Process with Dynamic Trust Using Uniformly Selected (τ, c) Pairs within Ranges $1 \leq c \leq 2$ and $0 \leq \tau \leq c$. Histograms depict the final consensus values (a) and final number of time steps (b) for 10,000 dynamic trust-based consensus simulations of the network in Figure 6.1.



(a)

(b)

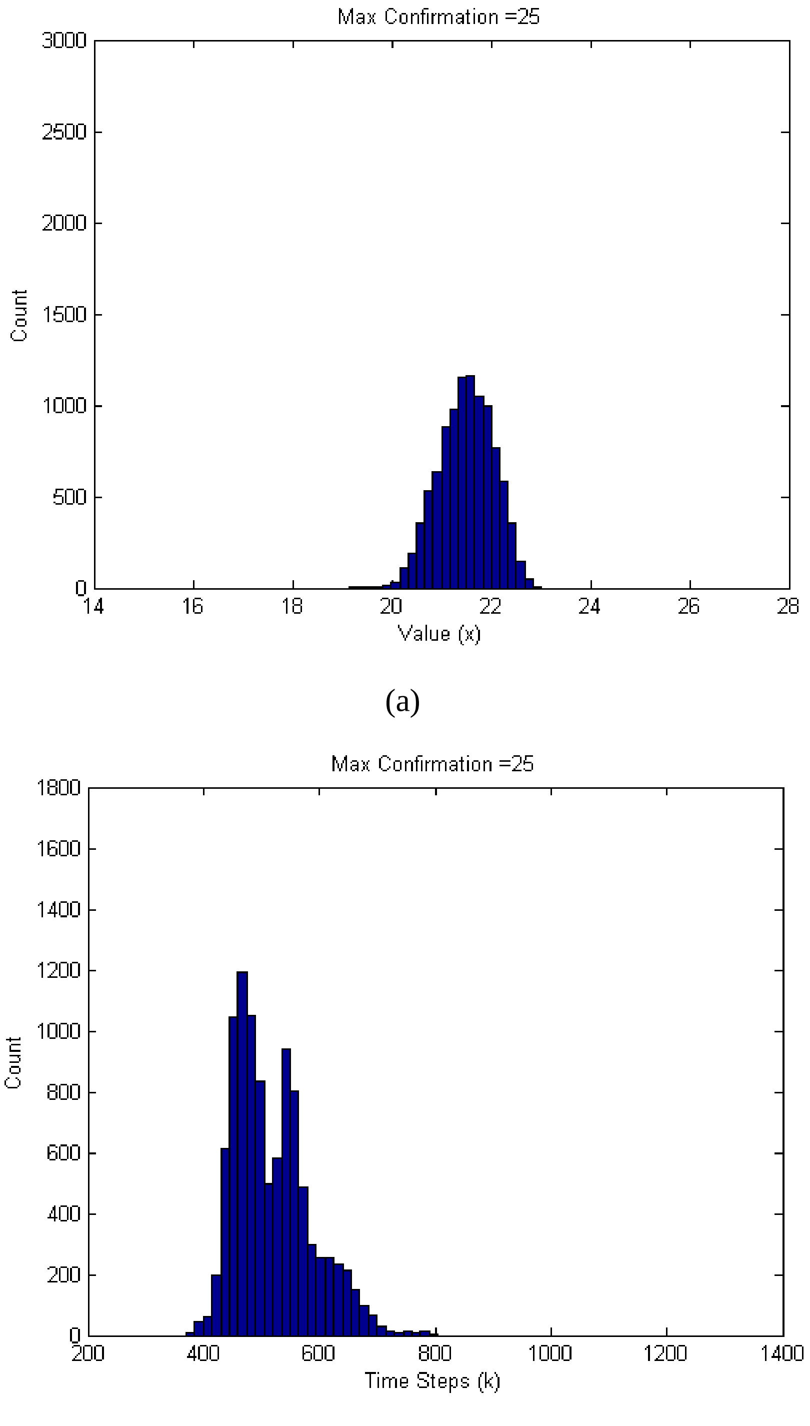
Figure 6.15. Histograms of the Consensus Process with Dynamic Trust Using Uniformly Selected (τ, c) Pairs within Ranges $1 \leq c \leq 5$ and $0 \leq \tau \leq c$. Histograms depict the final consensus values (a) and final number of time steps (b) for 10,000 dynamic trust-based consensus simulations of the network in Figure 6.1.



(a)

(b)

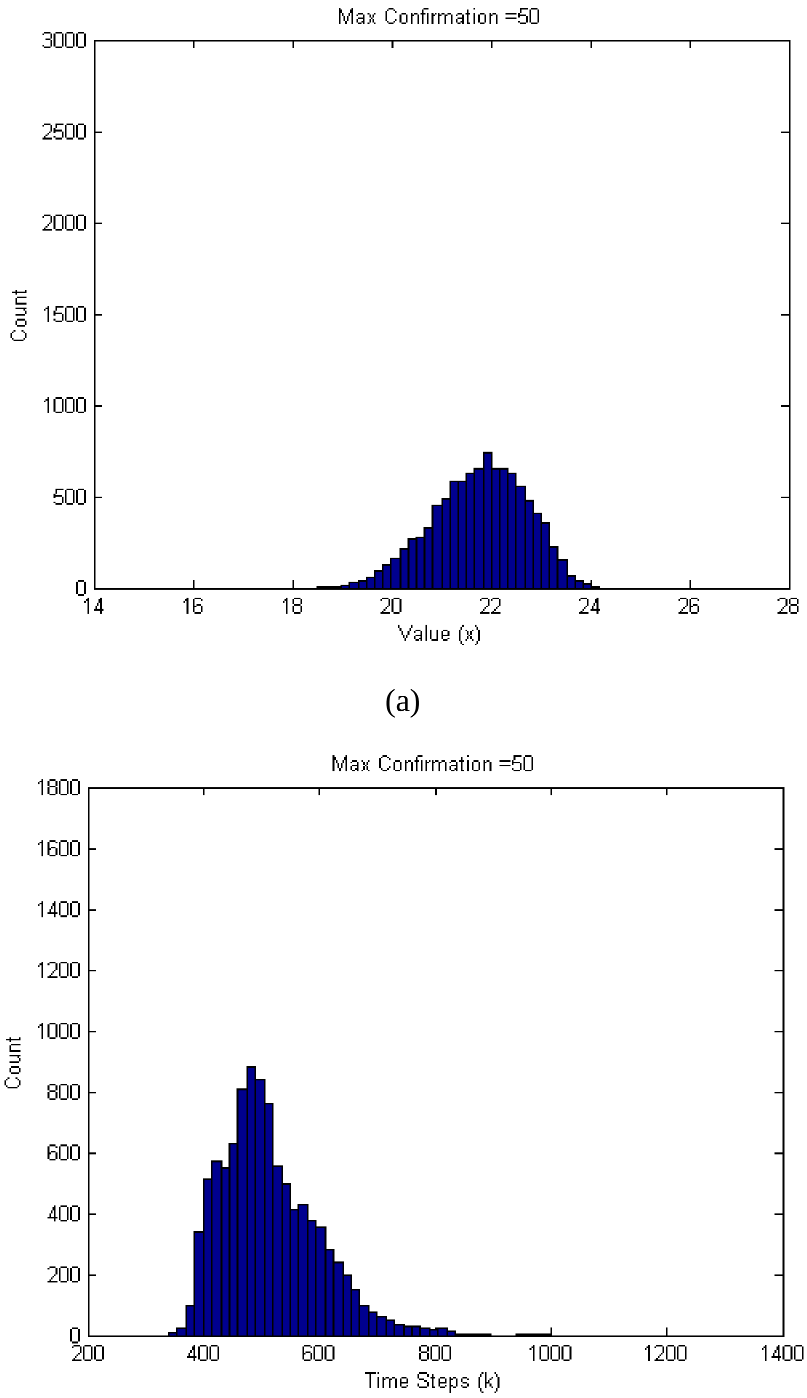
Figure 6.16. Histograms of the Consensus Process with Dynamic Trust Using Uniformly Selected (τ, c) Pairs within Ranges $1 \leq c \leq 10$ and $0 \leq \tau \leq c$. Histograms depict the final consensus values (a) and final number of time steps (b) for 10,000 dynamic trust-based consensus simulations of the network in Figure 6.1.



(a)

(b)

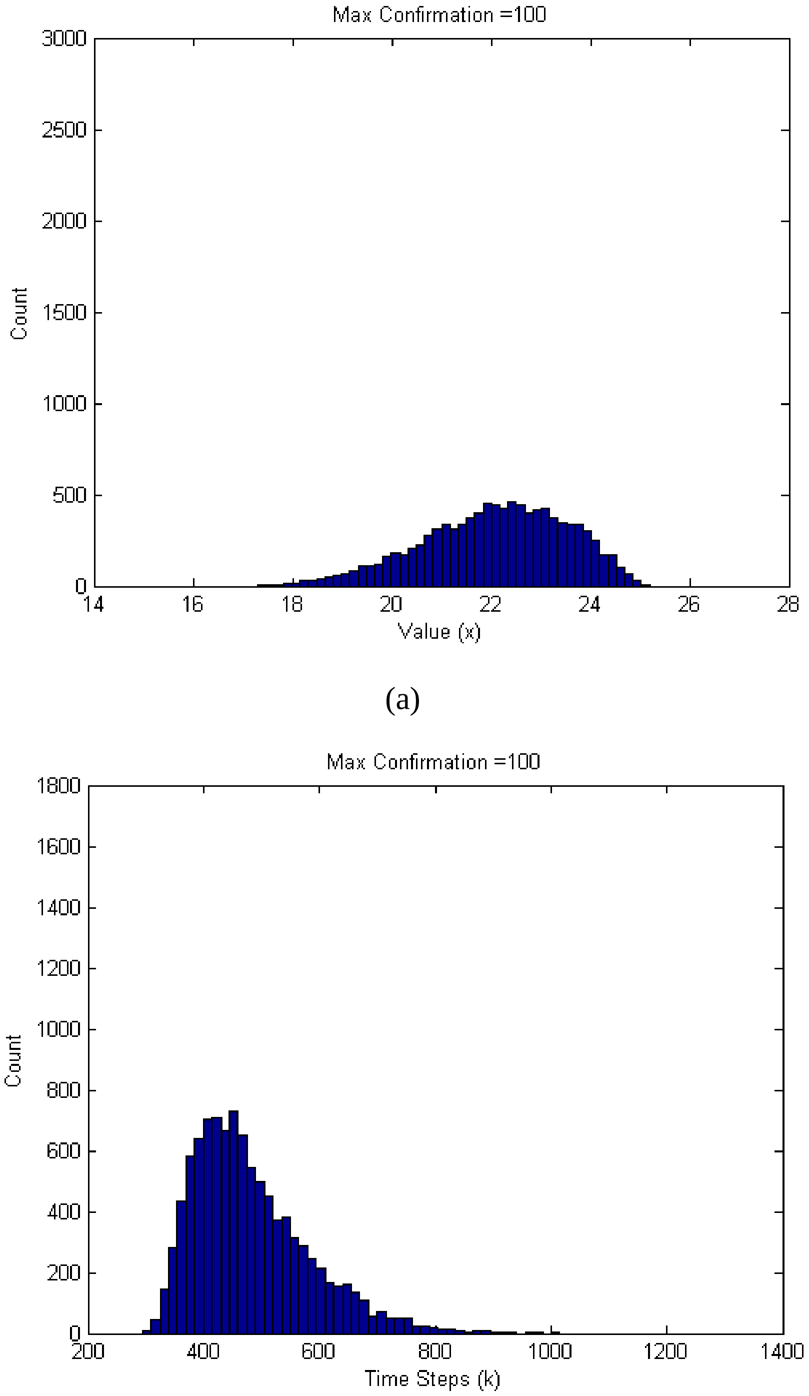
Figure 6.17. Histograms of the Consensus Process with Dynamic Trust Using Uniformly Selected (τ, c) Pairs within Ranges $1 \leq c \leq 25$ and $0 \leq \tau \leq c$. Histograms depict the final consensus values (a) and final number of time steps (b) for 10,000 dynamic trust-based consensus simulations of the network in Figure 6.1.



(a)

(b)

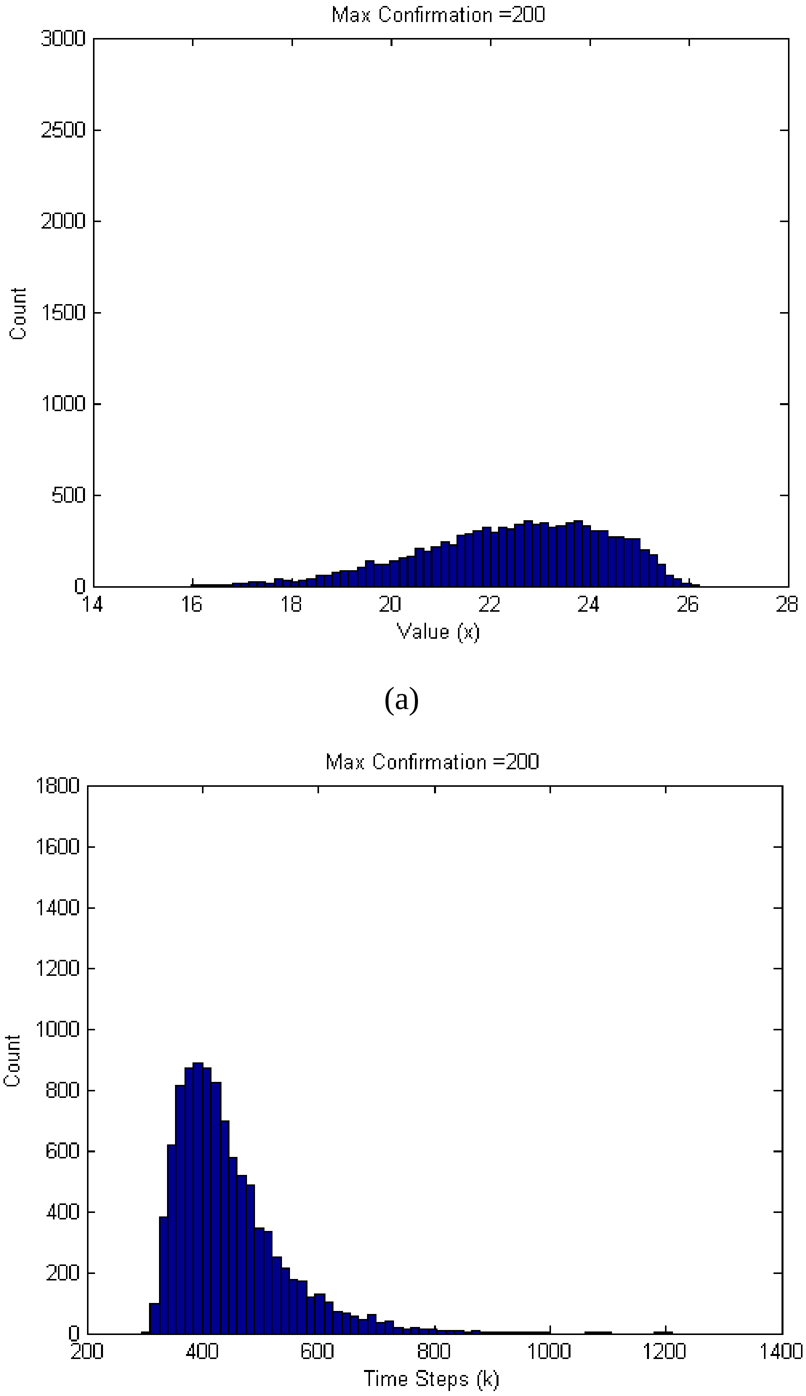
Figure 6.18. Histograms of the Consensus Process with Dynamic Trust Using Uniformly Selected (τ, c) Pairs within Ranges $1 \leq c \leq 50$ and $0 \leq \tau \leq c$. Histograms depict the final consensus values (a) and final number of time steps (b) for 10,000 dynamic trust-based consensus simulations of the network in Figure 6.1.



(a)

(b)

Figure 6.19. Histograms of the Consensus Process with Dynamic Trust Using Uniformly Selected (τ, c) Pairs within Ranges $1 \leq c \leq 100$ and $0 \leq \tau \leq c$. Histograms depict the final consensus values (a) and final number of time steps (b) for 10,000 dynamic trust-based consensus simulations of the network in Figure 6.1.



(a)

(b)

Figure 6.20. Histograms of the Consensus Process with Dynamic Trust Using Uniformly Selected (τ, c) Pairs within Ranges $1 \leq c \leq 200$ and $0 \leq \tau \leq c$. Histograms depict the final consensus values (a) and final number of time steps (b) for 10,000 dynamic trust-based consensus simulations of the network in Figure 6.1.

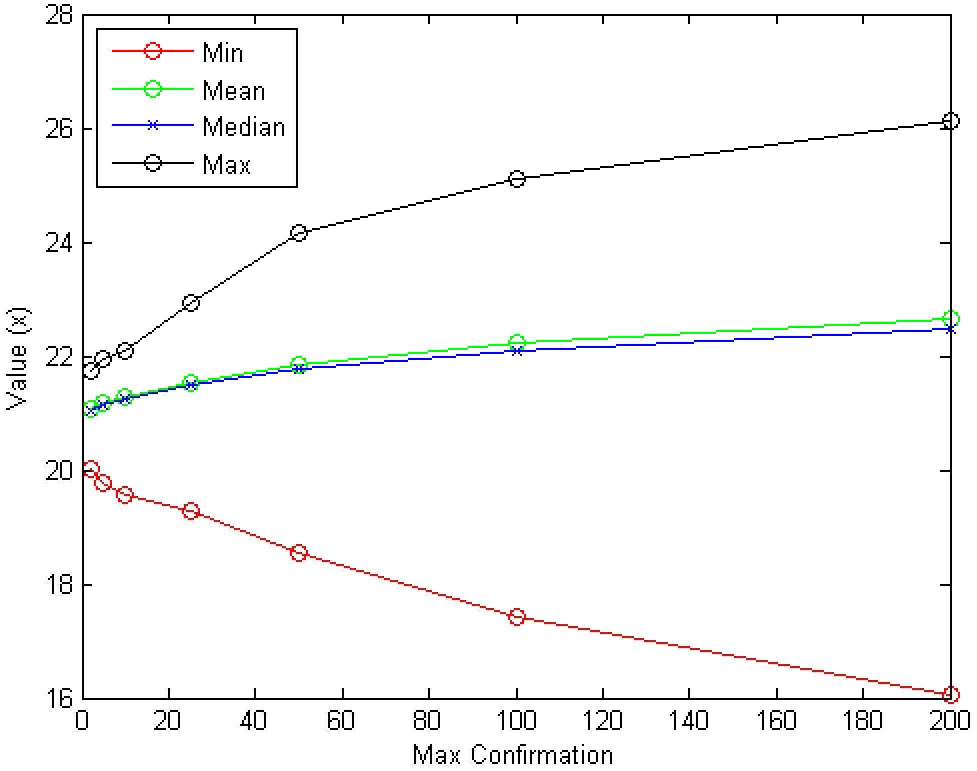


Figure 6.21. Histogram Statistics about the Final Consensus Value in Relation to the Max Confirmation. (This figure is presented in color; the black and white reproduction may not be an accurate representation.)

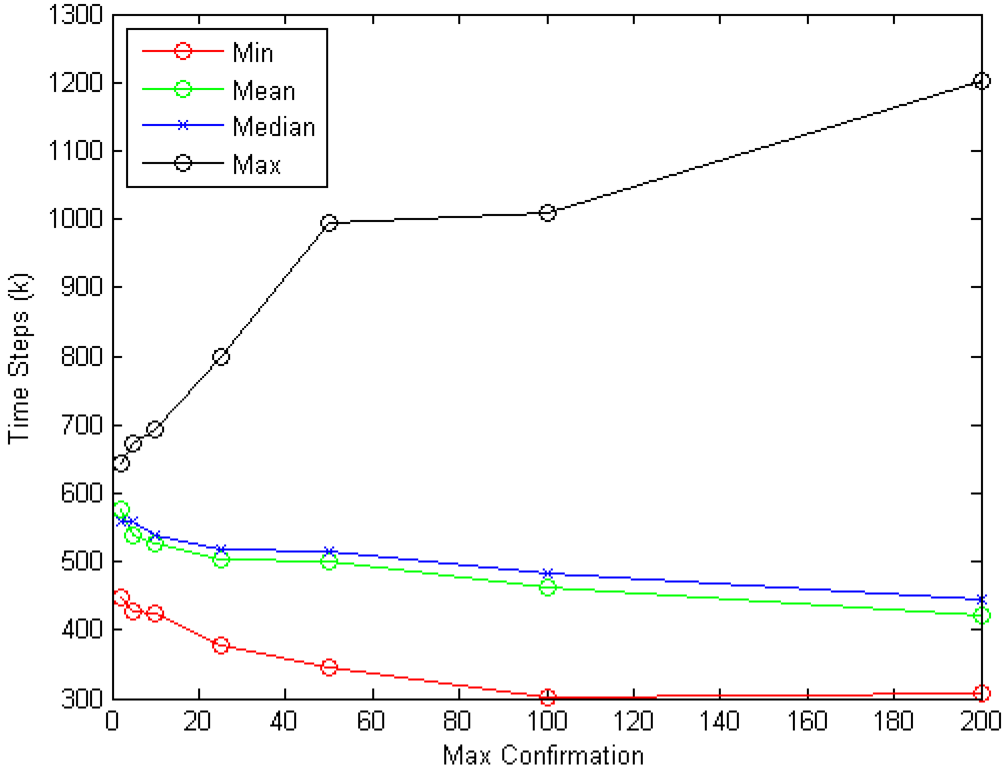


Figure 6.22. Histogram Statistics about the Final Consensus Time Steps in Relation to the Max Confirmation. (This figure is presented in color; the black and white reproduction may not be an accurate representation.)

Table 6.1.

Histogram Consensus Value Statistics

c_{max}	Min	Median	Mean	Std. Dev.	Max
2	20.0121	21.0558	21.0329	0.3815	21.7285
5	19.7620	21.1835	21.1552	0.2708	21.9353
10	19.5594	21.2754	21.2546	0.2908	22.0830
25	19.2939	21.5239	21.5071	0.5454	22.9218
50	18.5285	21.8441	21.7793	0.9304	24.1428
100	17.4383	22.2162	22.1037	1.4399	25.0953
200	16.0563	22.6648	22.4743	1.8385	26.1219

Table 6.2.

Histogram Consensus Time Step Statistics

c_{max}	Min	Median	Mean	Std. Dev.	Max
2	449	577	558.1428	54.5052	644
5	428	538	558.2354	59.9429	672
10	424	528	539.1214	62.7821	694
25	378	504	518.3130	66.1301	798
50	344	499	513.9163	84.3611	993
100	302	461	481.4641	99.6582	1010
200	308	422	444.5598	92.6045	1203

CHAPTER SEVEN

TRUST-BASED CONTROL FOR AUTONOMOUS CONVOY OPERATIONS

Synopsis

Autonomous convoy operations will likely be one of the early large-scale ground robotics missions to be executed by the U.S. Army within the next decade. However, the presence of many autonomous vehicles with their inherent exposure to cyber attacks introduces new trust-based vulnerabilities that previously did not exist. As such, the autonomous convoy mission represents a relevant and constrained application of the computational trust problem.

In this chapter, we develop and demonstrate a simulated trust-based controller for decentralized autonomous convoy operations. Section 7.1 analyzes the decentralized convoy using cooperative trust game theory. Section 7.2 describes the implementation of the trust-based controller within the framework of a convoy simulator. Section 7.3 analyzes three selected case studies using the trust-based controller in the simulation environment.

7.1 Convoy Trust Game Analysis for Decentralized Control

In Section 3.4, our 4-agent convoy trust game analysis showed that the hub-and-spoke communications network would establish the necessary connectivity to maximize the trust payoff between the leader and its followers to move together in a convoy (seen

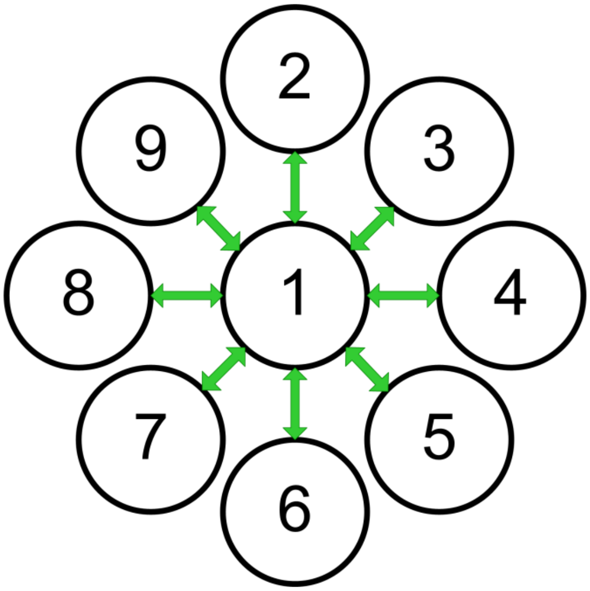


Figure 7.1. Bi-Directional Communications Hub-and-Spoke Graph.

in Figure 7.1). In essence, the leader (agent 1) would serve as the trusted third-party for all followers in the convoy, thereby allowing the convoy to maximize its trust synergy while minimizing its trust liability. For practical security reasons, one could envision the leader in such a network to be a human operator, who oversees the supervised autonomy of its robotic followers.

However, as with most centralized solutions, the primary drawback of the hub-and-spoke network is the bottleneck at the hub, or single point of failure. This drawback could make it difficult for the hub to handle high demand network traffic in a timely fashion, particularly if the network grows to be relatively large. Worse yet, temporary communication outages at the hub could generate mass confusion at the spokes, leading to unexpected consequences. In addition, the hub is particularly vulnerable to cyber attacks by motivated adversaries, given its high connectivity to every other node in the network. For these reasons, it is reasonable to explore the possibility of decentralized control for autonomous convoy operations.

As a prelude to the development of a trust-based controller, we perform a game theoretic analysis of a decentralized convoy using the cooperative trust game theory in Chapter Three. Our goal is this analysis is to understand the manner in which coalitions can form under this scenario and apply these insights to the trust-based controller in the next section.

7.1.1 Decentralized 4-Agent Convoy Trust Game

We begin with the same convoy scenario in Section 3.4.1 that models a four-agent convoy, $N = \{1,2,3,4\}$, which intends to move together in a single file. As before, we interpret the trust synergy to represent the agents in the coalition moving forward and the trust liability to represent the vulnerability of agents in the coalition to stop moving. The analysis will use the cooperative trust game model described by Equation 3.21.

For a decentralized formulation of the 4-agent convoy trust game, we must assume two conditions. First, we assume that the order of the agents in the convoy remains fixed; that is, no agent can alter its given position in the convoy file. Second, we assume that an agent i can only interact with another agent j if agent j is directly in front or behind agent i in the convoy file. In other words, agent i is restricted to only interact with agents that satisfy the condition $(i-1) \leq j \leq (i+1)$. As such, the values in $\mathbf{\Sigma}$ and $\mathbf{\Lambda}$ for a 4-agent convoy trust game will be a modified version of the matrices in Equation 3.23 to restrict coalition formation with ineligible agents.

$$\boldsymbol{\Sigma} = \begin{bmatrix} 0 & 2 & 0 & 0 \\ 2 & 0 & 3 & 0 \\ 0 & 3 & 0 & 4 \\ 0 & 0 & 4 & 0 \end{bmatrix} \quad \boldsymbol{\Lambda} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 0 & 2 & 0 & 2 \\ 0 & 0 & 3 & 0 \end{bmatrix} \tag{7.1}$$

Because of this restriction, the trust matrix $\boldsymbol{T}$ must conform to the mathematical structure in Equation 7.2.

$$\boldsymbol{T} = \left[\boldsymbol{T}_{ij}\right]_{|N|\times|N|} = \begin{cases} \boldsymbol{T}_{ij} = 1 & i = j \\ \boldsymbol{T}_{ij} \in [0,1] & (i-1) \le j \le (i+1) \\ \boldsymbol{T}_{ij} = 0 & \text{otherwise} \end{cases} \tag{7.2}$$

Therefore, the general form of the trust matrix $\boldsymbol{T}$ for the decentralized 4-agent convoy trust game is given in Equation 7.3. The graphical representation of this network is similarly shown in Figure 7.2.

$$\boldsymbol{T} = \begin{bmatrix} 1 & \boldsymbol{T}_{12} & 0 & 0 \\ \boldsymbol{T}_{21} & 1 & \boldsymbol{T}_{23} & 0 \\ 0 & \boldsymbol{T}_{32} & 1 & \boldsymbol{T}_{34} \\ 0 & 0 & \boldsymbol{T}_{43} & 1 \end{bmatrix} \tag{7.3}$$

Using Equation 3.21, we plot the payoff value surfaces for all valid agent pairs, namely $\{1,2\}, \{2,3\}, \{3,4\}$, by varying the trust values for the particular agent pair (in Figure 7.3). By inspection, we observe that the maximum positive value for agents 1 and 2 occurs when both agents fully trust each other (i.e. $\boldsymbol{T}_{12} = \boldsymbol{T}_{21} = 1$). We also observe that agent 2 and 3 will never form a coalition with each other, since their maximum payoff value is zero, even if they fully trust each other. The same observation can be made for agents 3 and 4 – the maximum payoff value never exceeds zero, implying that agents 3 and 4 have no incentive to form a coalition pair.

Using the results in Figure 7.3, we construct the trust matrices that produce the highest payoff value coalitions for the decentralized 4-agent convoy trust game in Equation 7.4.

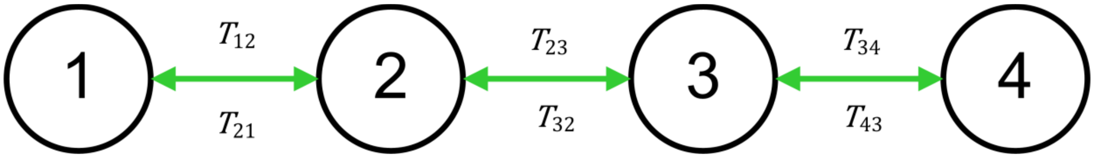
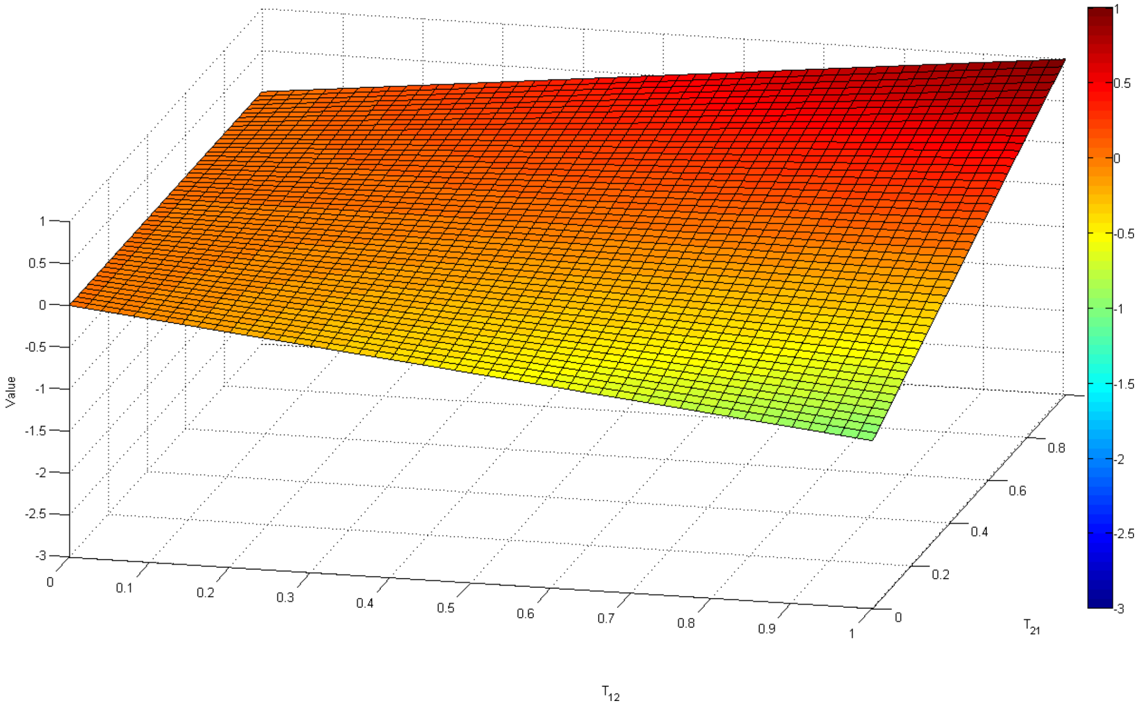
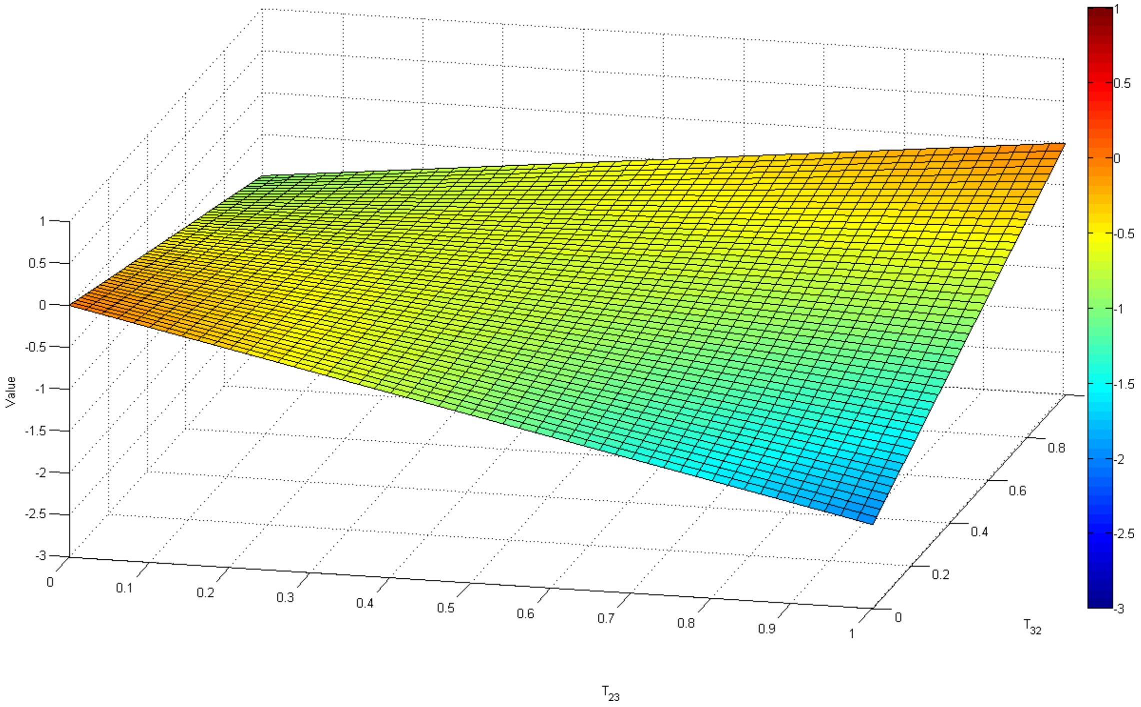


Figure 7.2. Four-Agent Convoy Network with First-Neighbor Local Communication.



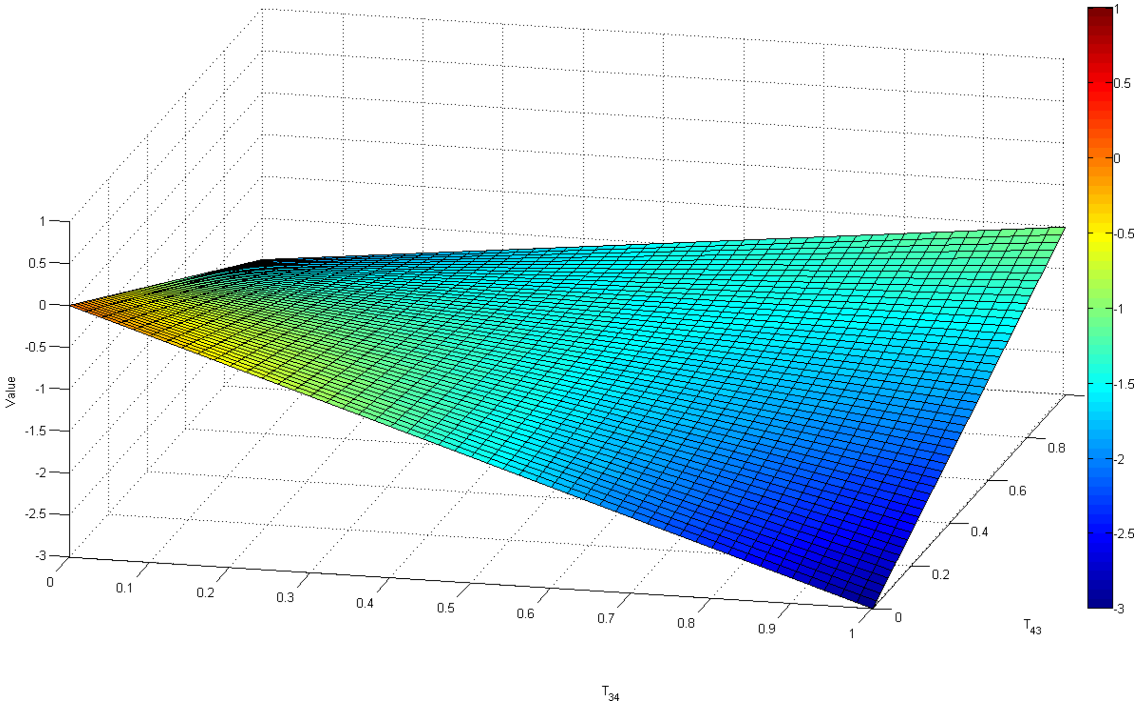
(a)

Figure 7.3. Payoff Value Surfaces for All Valid Agent Pairs, namely (a) {1,2}, (b) {2,3}, and (c) {3,4} (This figure is presented in color; the black and white reproduction may not be an accurate representation.)



(b)

Figure 7.3 – Continued



(c)

Figure 7.3 – Continued

Note that by inspection that both matrices produce the exact same payoff values for the decentralized 4-agent convoy trust game.

$$\boldsymbol{T}^{(1)} = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \boldsymbol{T}^{(2)} = \begin{bmatrix} 1 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{7.4}$$

Finally, we enumerate the maximum trust payoff values for each coalition.

$$v(\{1,2\}) = 1; \quad v(\{1,3\}) = 0; \quad v(\{1,4\}) = 0; \quad v(\{2,3\}) = 0;$$

$$v(\{2,4\}) = 0; \quad v(\{3,4\}) = 0; \quad v(\{1,2,3\}) = 1; \quad v(\{1,2,4\}) = 1;$$

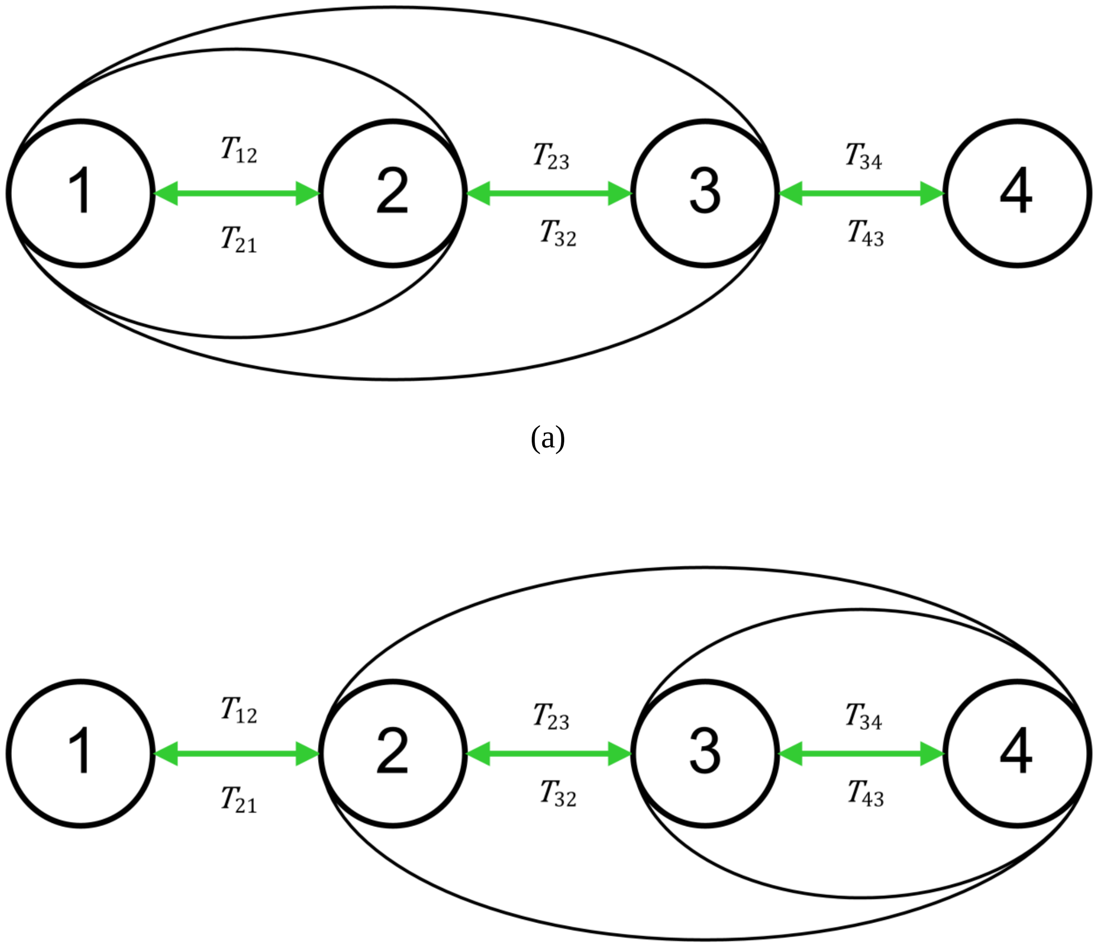
$$v(\{1,3,4\}) = 0; \quad v(\{2,3,4\}) = 0; \quad v(\{1,2,3,4\}) = 1;$$

7.1.2 Discussion

The results of the decentralized 4-agent trust game show that all four agents can form the grand coalition, provided that agent 1 and 2 are both members of the coalition. However, this conclusion comes with a caveat since the trust payoff value does not grow higher than 1, no matter how many additional agents are included in the coalition with agents 1 and 2. Recall that the trust games throughout this dissertation assume transferable utility, where payoffs are distributed freely among coalition members. Thus, any agent that joins the coalition of agent 1 and 2 will need to receive positive non-zero utility in order to remain in the coalition. This would mean that agent 1 or agent 2 (or both) would need to transfer a portion of the total payoff to the additional agents, thereby diluting their own individual payoffs. Thus, while theoretically possible, it is practically implausible that either agent 1 or agent 2 would want to transfer any trust payoff utility to an additional dummy agent.

Our analysis exposes a conundrum of sorts, in that it appears that the best that decentralized control can amount to is cooperation between the first two agents and no cooperation between any of the other agents. However, this conundrum can be remedied if we consider the individual frame of reference for each agent in the convoy. By doing so, we can reduce a single decentralized convoy trust game to many 2-agent convoy trust games. We can do this because we know that each agent can only interact with their immediate leader and their immediate follower, and no other agent. Furthermore, we also know that each agent has the knowledge that there are other agents in the convoy that are unobservable, but may influence the behaviors of its immediate neighbors. Thus, if each agent can consider their immediate leaders as surrogates for the system of agents in front of it and their immediate followers as surrogates for the system of agents behind it, then we can formulate the decentralization control problem in a relativistic fashion, where an agent can view itself simultaneously as a follower of a super-agent leader or a leader of a super-agent follower. Figure 7.4 graphically depicts this concept.

To verify our understanding of this theoretical formulation, let us consider the anecdotal evidence found in our experiences with automobile traffic. Drivers in traffic lanes (coalitional convoys) can often only reliably observe the behavior of the vehicle directly in front and behind them, just as in our relativistic formulation. And using the observations of their immediate neighbors, drivers can estimate the payoff value of their traffic lane. If the immediate leader is driving at the posted speed limit, then the driver can reasonably conclude that vehicles in front of the immediate leader are also able to drive the posted speed limit, implying a high coalitional value for the current traffic



(a)

(b)

Figure 7.4. Surrogate Perspective of a 4-Agent Convoy Network with First-Neighbor Local Communication. A graphical depiction of a 4-agent network that only allows local communication with first-neighbors in a convoy, where agents view their immediate leaders as surrogates to systems of vehicles in front of them (a) and their immediate followers as surrogates to systems of vehicles behind them (b). By doing so, any agent interaction in a decentralized convoy can be analyzed as a 2-agent convoy trust game between a super-agent leader and a super-agent follower.

lane. However, if a driver observes that the immediate leader is moving significantly below the posted speed limit for a long period of time, then he may reasonably conclude that other vehicles in front of the immediate leader are also moving significantly below the posted speed limit, thereby implying a traffic jam. A driver in a traffic jam situation will unconsciously begin gauging the coalitional value of the traffic jam by considering his level of trust in the collection of vehicles in front of him to move forward at a more reasonable speed. A driver may factor in observations of its immediate neighbors, the traffic flow of adjacent traffic lanes, and the traffic reports on the radio to improve the accuracy of the coalitional payoff estimate. If the coalitional payoff becomes too low in a driver's current lane, then the driver may choose to disband from his current lane and attempt to join another traffic coalition (lane) with a higher payoff value.

We can also consider the anecdotal case of driving in a funeral procession. In this case, the driver may be the local leader of a collection of follower vehicles. If the immediate follower is maintaining a reasonable following distance from the driver, then the driver can reasonably conclude that the vehicles behind the immediate follower are also maintaining reasonable following distances with each other. However, if the distance of the immediate follower is unusually large, then the driver may reasonably conclude that one or more of the follower vehicle behind him cannot be trusted to keep up with the current pace of the procession. As such, the driver may decide to slow down in order to decrease the following distance between him and his immediate follower to maintain the stability of the procession. It is highly likely that the driver's immediate leader would notice the driver's reduction in speed and also reasonably slow down for the same reasons as the driver. Through this process, one could see how information about the procession stability would eventually propagate to the hearse driver, the global leader in the procession.

7.2 Trust-Based Convoy Simulator

The anecdotal cases of the traffic jam and funeral procession in the previous section highlight how information is able to propagate in both directions in a convoy without the need of a centralized hub. In addition, the decentralized convoy trust game analysis helped us provide the mathematical justification for the correct way to view other vehicles in a decentralized convoy – as surrogates for a larger system of vehicles in front or behind them. Building on these findings, we are now able to implement a simple, yet valid trust-based vehicle controller for decentralized convoy behavior. This section describes the implementation of the trust-based controller within the framework of a Matlab convoy simulator.

7.2.1 Simulator Physics and Environment

The trust-based convoy simulator models the environment as an obstacle-free, frictionless, flat 2-dimensional environment in the xy plane, which we refer to as the world. We purposely chose to keep the world as simple as reasonably permissible in order to ensure that the focus of the simulations remain on the trust interactions between the convoy vehicles.

Let W be the non-empty set of given waypoints in the world. As such, we define the waypoint path as a loop, described by the infinite sequence $\left[W_{(\omega \bmod |W|)}\right]_{\omega=1}^{\infty}$.

Let N be the non-empty set of all vehicles in the world. Each vehicle $i \in N$ is modeled as a single-point particle at location $\boldsymbol{x}_i(k)$ at time step $k \in \mathbb{N}$ (in seconds), with vehicle characteristics that include: mass m_i in kilograms, minimum following distance d_i in meters, maximum speed s_i in meters per second, maximum impulse force p_i in Newtons per second, sensor range r_i in meters, and sensor horizontal field of view θ_i in degrees. The initial positions of each vehicle are placed along the vector between the first and last waypoints on the path, each spaced out by the length of their respective sensor range r_i, facing forward in the direction $\boldsymbol{h}_i$ toward the first waypoint.

$$\boldsymbol{h}_i = \frac{W_{(1)} - W_{(|W|)}}{\left\| W_{(1)} - W_{(|W|)} \right\|} \quad \forall i \in N \tag{7.5}$$

$$\boldsymbol{x}_1(0) = W_{(1)} - (r_1 \boldsymbol{h}_1) \tag{7.6}$$

$$\boldsymbol{x}_{i+1}(0) = \boldsymbol{x}_i(0) - (r_{i+1} \boldsymbol{h}_1) \tag{7.7}$$

After initialization, each vehicle is allowed to apply an impulse force vector, $\dot{\boldsymbol{p}}_i$, in any direction within the xy plane for locomotion within the world. Successive impulse forces accumulate over time to give the vehicle its linear momentum $\boldsymbol{p}_i(k)$.

$$\boldsymbol{p}_i(0) = 0 \tag{7.8}$$

$$\boldsymbol{p}_i(k+1) = \boldsymbol{p}_i(k) + \dot{\boldsymbol{p}}_i \tag{7.9}$$

In order to simulate allowable linear accelerations between each time step, each vehicle constrains the magnitude of its desired impulse force by a maximum impulse force scalar, p_i, such that $\|\dot{\boldsymbol{p}}_i\| \leq p_i$. To do this, we define a scaling function to determine the multiple needed to adjust the magnitude of a desired impulse force. Constraints such as this are often implemented in actual drive-by-wire systems in order to guarantee that the desired drive signals fall within the appropriate vehicle operating limits.

$$f(\boldsymbol{q}, q) = \begin{cases} \frac{q}{\|\boldsymbol{q}\|} & \|\boldsymbol{q}\| > q \\ 1 & \|\boldsymbol{q}\| \leq q \end{cases} \tag{7.10}$$

Using the scaling function, we may now calculate the allowable vehicle impulse force using Newton's second law of motion. Note, however, that the calculation depends on a target velocity $\boldsymbol{u}_i$, which is derived from the trust-based vehicle controller. The details of how the vehicle controller determines $\boldsymbol{u}_i$ will be discussed in the next section, but for the time being, assume that $\boldsymbol{u}_i$ is given.

$$\widehat{\boldsymbol{p}}_i = m_i \boldsymbol{u}_i - \boldsymbol{p}_i(k) \tag{7.11}$$

$$\dot{\boldsymbol{p}}_i = f(\widehat{\boldsymbol{p}}_i, p_i)\widehat{\boldsymbol{p}}_i \tag{7.12}$$

Next, using the linear momentum $\boldsymbol{p}_i(k+1)$, we calculate the new velocity $\dot{\boldsymbol{x}}_i$. We also track the heading $\boldsymbol{h}_i$ of the vehicle i independently so that we may know which direction i is facing when $\dot{\boldsymbol{x}}_i = \boldsymbol{0}$.

$$\dot{\boldsymbol{x}}_i = \frac{\boldsymbol{p}_i(k+1)}{m_i} \tag{7.13}$$

$$\boldsymbol{h}_i = \dot{\boldsymbol{x}}_i \quad \dot{\boldsymbol{x}}_i \neq \boldsymbol{0}$$

Finally, we update the current position, $\boldsymbol{x}_i(k)$, through vector addition.

$$\boldsymbol{x}_i(k+1) = \boldsymbol{x}_i(k) + \dot{\boldsymbol{x}}_i \tag{7.14}$$

7.2.2 Trust-Based Vehicle Controller

The purpose of the simulated trust-based vehicle controller is to calculate the target velocity vector for vehicle i, namely $\boldsymbol{u}_i$. The calculation depends on information acquired through active sensing and/or passive listening. Active sensing may only occur within the area described by the sector originating from $\boldsymbol{x}_i(k)$, defined by r_i and θ_i and oriented such that the forward vehicle direction vector bisects the sector. Within the sensing sector, a vehicle may detect the identity of other vehicles along with their respective absolute positions and velocities. Formally, we describe N_i as the set of first-neighbors of vehicle i in the sensor range (from Equation 5.11), and their positions and velocities as $\boldsymbol{x}_j(k)$ and $\dot{\boldsymbol{x}}_j(k)$, respectively, where $j \in N_i$. A vehicle i may also broadcast information to each vehicle j, such as its own identity i, velocity $\dot{\boldsymbol{x}}_i(k)$, and current waypoint $\boldsymbol{w}_i = W_{(\omega_i \bmod |W|)}$, where ω_i is vehicle i's current index on the waypoint path. We assume that each vehicle j will always receive broadcasted messages with no interference from another vehicle or world.

The vehicle controller is allowed to determine a target velocity vector $\boldsymbol{u}_i$ to point in any direction within the xy plane of the world. However, the magnitude of the target velocity may not exceed the scalar s_i, the maximum speed for vehicle i. To enforce this constraint, we use the scaling function in Equation 7.10 to determine the multiple needed to adjust the magnitude of a desired velocity vector $\hat{\boldsymbol{u}}_i$ to an allowable operating limit.

$$\boldsymbol{u}_i = f(\hat{\boldsymbol{u}}_i, s_i)\hat{\boldsymbol{u}}_i \tag{7.15}$$

To decide on the appropriate direction for the desired velocity, the trust-based controller chooses its target according to the following priority:

1. When N_i is not empty, then the target is the location of the nearest trusted leader in the sensor range, $\boldsymbol{x}_\ell(k)$, such that $\ell \in N_i$ and $\|\boldsymbol{x}_i(k) - \boldsymbol{x}_\ell(k)\| = \min_{j \in N_i}\left(\|\boldsymbol{x}_i(k) - \boldsymbol{x}_j(k)\|\right)$.
2. In the absence of a trusted leader within range, the target is a location that is perceived as acceptable to all trusted followers $j \in N \backslash i$, such that $i \in N_j$.
3. In the absence of both trusted leaders and trusted followers, the target is a location determined by the default vehicle control.

7.2.2.1 Following a Trusted Leader

The control used by vehicle i to follow a trusted leader ℓ is based on adaptive cruise control. Essentially, vehicle i attempts to match the speed of vehicle ℓ at the minimum following distance d_i if the leader trust value $\boldsymbol{T}_{i\ell}^{(\mathcal{L})}$ is greater than the leader trust threshold $t_i^{(\mathcal{L})}$. The value for $t_i^{(\mathcal{L})}$ is preset before the start of a simulation and remains fixed throughout the entire simulation duration. The value for $\boldsymbol{T}_{i\ell}^{(\mathcal{L})}$ is calculated using the RoboTrust model described in Equation 5.10.

RoboTrust requires a binary observation history $\boldsymbol{z}_{i\ell}$ in order to calculate the trust value $\boldsymbol{T}_{i\ell}^{(\mathcal{L})}$. This history is generated through the use of acceptance functions, which describe the acceptable portions of a feature space for a particular context. In our simulator, each vehicle can chose one of a possible six acceptance functions to evaluate the behavior of a leader.

1. **Closest Leader Always Wrong**. In this context, the acceptance function always outputs an unacceptable value, regardless of the observed behavior

of the leader ℓ. This context is used to ensure that a vehicle never follows another vehicle.

$$\boldsymbol{z}_{i\ell}(k) = 0 \tag{7.16}$$

2. **Closest Leader Always Right**. In this context, the acceptance function always outputs an acceptable value, regardless of the observed behavior of the leader ℓ. This context is used to ensure that a vehicle never stops following a lead vehicle once it has decided to follow that lead vehicle.

$$\boldsymbol{z}_{i\ell}(k) = 1 \tag{7.17}$$

3. **Closest Leader Is Moving**. In this context, the acceptance function will output an acceptable value if the leader ℓ has a speed greater than zero, regardless of the direction it is traveling.

$$\boldsymbol{z}_{i\ell}(k) = \begin{cases} 1 & \|\dot{\boldsymbol{x}}_\ell\| > 0 \\ 0 & \text{otherwise} \end{cases} \tag{7.18}$$

4. **Closest Leader Going To My Waypoint**. In this context, the acceptance function outputs an acceptable value if the leader ℓ is closer to waypoint $\boldsymbol{w}_i$ now than at a time step before.

$$\boldsymbol{z}_{i\ell}(k) = \begin{cases} 1 & \|\boldsymbol{w}_i - \boldsymbol{x}_\ell(k)\| < \|\boldsymbol{w}_i - \boldsymbol{x}_\ell(k-1)\| \\ 0 & \text{otherwise} \end{cases} \tag{7.19}$$

Note that ideally, one could simply check that $\boldsymbol{w}_i = \boldsymbol{w}_\ell$ for an acceptable output from the acceptance function. But because ℓ cannot directly communicate with i, this context is used to infer from the behavior of ℓ that i and ℓ share the same current waypoint.

5. **Closest Leader Heading To My Waypoint**. In this context, the acceptance function outputs an acceptable value if the directional heading of leader ℓ is aligned towards waypoint $\boldsymbol{w}_i$ within 0.5 radians.

$$\boldsymbol{v} = \boldsymbol{w}_i - \boldsymbol{x}_\ell(k) \tag{7.20}$$

$$\boldsymbol{z}_{i\ell}(k) = \begin{cases} 1 & \min(|\tan^{-1}\boldsymbol{h}_\ell - \tan^{-1}\boldsymbol{v}|, |\tan^{-1}\boldsymbol{v} - \tan^{-1}\boldsymbol{h}_\ell|) \leq 0.5 \\ 0 & \text{otherwise} \end{cases} \tag{7.21}$$

This context is used to infer that $\boldsymbol{w}_i = \boldsymbol{w}_\ell$, even if ℓ is not moving. When moving, however, this context can be used to detect drifting, even if ℓ is successively shortening the distance between itself and $\boldsymbol{w}_i$.

6. **Closest Leader Fast Enough**. In this context, the acceptance function outputs an acceptable value if vehicle i is able to move faster than 50% of its maximum speed when following the leader ℓ.

$$\boldsymbol{z}_{i\ell}(k) = \begin{cases} 1 & \|\dot{\boldsymbol{x}}_i\| > \frac{s_i}{2} \\ 0 & \text{otherwise} \end{cases} \tag{7.22}$$

This context is different than all of the previous contexts in the sense that it does not require the measurement of any feature of ℓ to evaluate the trustworthiness of ℓ. Rather, the context evaluates whether i is performing within the desired operating limits in response to following ℓ.

If it is the case that $\boldsymbol{T}_{i\ell}^{(\mathcal{L})} > t_i^{(\mathcal{L})}$, then we may proceed with determining the desired velocity vector $\hat{\boldsymbol{u}}_i$. First, we determine the vector $\boldsymbol{v}_{i\ell}$ to discover both the direction toward ℓ as well as the following distance between i and ℓ.

$$\boldsymbol{v}_{i\ell} = \boldsymbol{x}_\ell(k) - \boldsymbol{x}_i(k) \tag{7.23}$$

Next, we determine the desired following speed of vehicle i, namely $\hat{s}_i$. This is done by regulating the value of vehicle ℓ's speed, $\|\dot{\boldsymbol{x}}_\ell\|$, by the ratio of the following distance $\|\boldsymbol{v}_{i\ell}\|$ and the minimum following distance d_i. If $\|\boldsymbol{v}_{i\ell}\| = d_i$, then the ratio is 1 so we would expect the speed of i and ℓ to be exactly the same. If $\|\boldsymbol{v}_{i\ell}\| > d_i$, then vehicle i needs to catch up to ℓ by moving faster than ℓ. Conversely, if $\|\boldsymbol{v}_{i\ell}\| < d_i$, then vehicle i needs to slow down since it has violated the set minimum following distance.

$$\hat{s}_i = \left(\frac{\|\boldsymbol{v}_{i\ell}\|}{d_i}\right)\|\dot{\boldsymbol{x}}_\ell\| \tag{7.24}$$

Finally, we set the desired velocity vector $\widehat{\boldsymbol{u}}_i$ by multiplying the unit vector of $\boldsymbol{v}_{i\ell}$ by $\hat{s}_i$.

$$\widehat{\boldsymbol{u}}_i = \left(\frac{\hat{s}_i}{\|\boldsymbol{v}_{i\ell}\|}\right)\boldsymbol{v}_{i\ell} \quad \boldsymbol{T}_{i\ell}^{(\mathcal{L})} > t_i^{(\mathcal{L})} \tag{7.25}$$

It should be noted, however, that by substituting Equation 7.24 into Equation 7.25, it is possible to completely eliminate the need to use the following distance $\|\boldsymbol{v}_{i\ell}\|$, thereby skipping the step of calculating the desired following speed $\hat{s}_i$ separately.

$$\widehat{\boldsymbol{u}}_i = \left(\frac{\|\dot{\boldsymbol{x}}_\ell\|}{d_i}\right)\boldsymbol{v}_{i\ell} \quad \boldsymbol{T}_{i\ell}^{(\mathcal{L})} > t_i^{(\mathcal{L})} \tag{7.26}$$

7.2.2.2 Leading a Trusted Follower

The purpose of this control is to ensure that a trusted follower $j \in N\backslash i$, such that $i \in N_j$, is trusted to continue to follow i. This can only happen if i is trusted to be a good leader from the perspective of j. However, i has no ability to sense or observe j directly to accurately evaluate this. Thus, the control to lead j depends on i's evaluation of the received communication from j. If vehicle i evaluates that j is satisfied with its leadership, then it will continue with its own control. Otherwise, i will temporarily adjust its control to satisfy the perceived desires of j, provided that the follower trust value $\boldsymbol{T}_{ij}^{(\mathcal{F})}$ is greater than the follower trust threshold $t_i^{(\mathcal{F})}$. The value for $t_i^{(\mathcal{F})}$ is preset before the start of a simulation and remains fixed throughout the entire simulation duration. The value for $\boldsymbol{T}_{ij}^{(\mathcal{F})}$ is calculated using the RoboTrust model described in Equation 5.10. Should $\boldsymbol{T}_{ij}^{(\mathcal{F})}$ fall below the threshold $t_i^{(\mathcal{F})}$, then i will assume that j is not trusted to follow, and therefore, i will not adjust its control to the perceived desires of j.

RoboTrust requires a binary observation history $\boldsymbol{z}_{ij}$ in order to calculate the trust value $\boldsymbol{T}_{ij}^{(\mathcal{F})}$. This history is generated through the use of acceptance functions, which describe the acceptable portions of a feature space for a particular context. In our simulator, each vehicle can choose one of a possible three acceptance functions to evaluate the desires of a follower.

1. **Closest Follower Dislikes Everything**. In this context, the acceptance function always outputs an unacceptable value, regardless of the information received from follower j. It pessimistically assumes that j can never be trusted to follow, even if the information i is receiving indicates that it is.

$$\boldsymbol{z}_{ij}(k) = 0 \tag{7.27}$$

2. **Closest Follower Likes My Waypoint**. In this context, the acceptance function outputs an acceptable value when both i and j have the same waypoint goal.

$$\boldsymbol{z}_{ij}(k) = \begin{cases} 1 & \boldsymbol{w}_i = \boldsymbol{w}_j \\ 0 & \text{otherwise} \end{cases} \tag{7.28}$$

3. **Closest Follower Likes My Heading**. In this context, the acceptance function outputs an acceptable value when the heading of j is well aligned with the heading of i.

$$\boldsymbol{z}_{ij}(k) = \begin{cases} 1 & \min\left(\left|\tan^{-1}\boldsymbol{h}_j - \tan^{-1}\boldsymbol{h}_i\right|, \left|\tan^{-1}\boldsymbol{h}_i - \tan^{-1}\boldsymbol{h}_j\right|\right) \leq \dfrac{\pi}{18} \\ 0 & \text{otherwise} \end{cases} \tag{7.29}$$

If it is the case that $\boldsymbol{T}_{ij}^{(\mathcal{F})} > t_i^{(\mathcal{F})}$, but the current observation is $\boldsymbol{z}_{ij}(k) = 0$, then we determine the desired velocity vector $\widehat{\boldsymbol{u}}_i$ by directing vehicle i toward $\boldsymbol{w}_j$. We must be mindful, however, that there could be multiple followers, some of whom may be both trusted and satisfied. As such, our control rule uses the current observation of each follower as a way to filter out trusted, satisfied followers from the trusted, unsatisfied followers.

$$\widehat{\boldsymbol{u}}_i = \sum_{j \in N \setminus i} \left(1 - \boldsymbol{z}_{ij}(k)\right)\left(\boldsymbol{w}_j - \boldsymbol{x}_i(k)\right) \quad \boldsymbol{T}_{ij}^{(\mathcal{F})} > t_i^{(\mathcal{F})}, i \in N_j \tag{7.30}$$

7.2.2.3 Default Vehicle Control

The default vehicle control sets the behavior of each vehicle in the absence of both a trusted leader and a trusted, unsatisfied follower. At a high level, each vehicle is considered to be in one of two states: active and halted. The vehicle controller can set its desired state through the use of a vehicle state parameter, $v_i \in \mathbb{Z}$. If $v_i \geq 1$, then the vehicle controller desires the vehicle to be in an active state, in which it is executing its default control law. If $v_i \leq 0$, then the vehicle controller desires the vehicle to be in a halted state for precisely $|v_i|$ seconds and then transitions back to an active state. A vehicle is considered to be halted only when the linear momentum $\boldsymbol{p}_i(k)$ is zero, and therefore, the amount of time required to decelerate to zero momentum is not subtracted from the total halt time of $|v_i|$ seconds. Formally, we update the vehicle state parameter at each time step as follows to track the remaining halt time.

$$v_i^* = v_i \tag{7.31}$$

$$v_i = v_i^* + 1 \quad v_i^* < 1, \boldsymbol{p}_i(k) = 0 \tag{7.32}$$

The convoy simulator provides several default vehicle control schemes, some of which are purposely designed to be abnormal.

1. **Always Moving**. In this control scheme, the vehicle i moves from waypoint to waypoint along its path without ever halting. As such, the desired velocity vector always points towards its current waypoint $\boldsymbol{w}_i$.

   $$\widehat{\boldsymbol{u}}_i = \boldsymbol{w}_i - \boldsymbol{x}_i(k) \tag{7.33}$$

   A new waypoint is selected only when $\boldsymbol{x}_i(k)$ falls within the circle defined by its center at $\boldsymbol{w}_i$ and radius w_i. The waypoint radius w_i is dynamic and changes in length according to a vehicle's linear momentum and maximum impulse force. This is done to minimize the effects of overshooting or undershooting the waypoint when targeting the next waypoint, regardless of the control scheme or dynamics of different vehicles. We set the length of the radius to an estimate of the stopping distance of the vehicle, given its momentum force and maximum impulse force. This estimate can be calculated recursively in the following manner.

$$R_i(p) = \begin{cases} \dfrac{p}{m_i} & p \leq p_i \\ \dfrac{p}{m_i} + R_i(p - \ p_i) & p > p_i \end{cases} \tag{7.34}$$

$$w_i = R_i(\|\boldsymbol{p}_i(k)\|) \tag{7.35}$$

2. **Follow Only**. In this control scheme, the vehicle's active state is set to be halted indefinitely. This essentially means that the vehicle can only move forward if it follows a trusted leader using the control law from Equation 7.26.

$$\widehat{\boldsymbol{u}}_i = \boldsymbol{0} \tag{7.36}$$

3. **Stop and Go**. In this control scheme, the vehicle i moves from waypoint to waypoint along its path, only halting when it reaches its maximum speed s_i. There are three variations of this control scheme, differing only by the desired duration of the halt time: 0 seconds, 10 seconds, and 30 seconds. If we let $k_i^* \in \mathbb{N}$ be the desired halt time for this control scheme, then our control is described as follows.

$$v_i = -k_i^* \quad \|\dot{\boldsymbol{x}}_i(k)\| \geq s_i) \tag{7.37}$$

$$\widehat{\boldsymbol{u}}_i = \begin{cases} \boldsymbol{w}_i - \boldsymbol{x}_i(k) & v_i \geq 1 \\ \boldsymbol{0} & v_i < 1 \end{cases} \tag{7.38}$$

4. **Stop at Waypoint**. In this control scheme, the vehicle i moves from waypoint to waypoint along its path, only halting when it switches to a new waypoint. This, in effect, causes the vehicle to stop at the waypoint it was

previously targeting. This control scheme uses the control law in Equation 7.38 and sets $v_i = 0$ when a new waypoint is selected.

5. **Steering (Wide)**. In this control scheme, the vehicle i moves from waypoint to waypoint along its path. However, the velocity vector pointing toward the current waypoint is augmented with an orthogonal steering velocity vector. This causes the vehicle to take a wide curved path towards the waypoint rather than a direct straight line path. We calculate the orthogonal vector using the Gram-Schmidt process, a method for orthonormalising independent vectors in an inner product space.

$$\boldsymbol{v}_i^{(1)} = \boldsymbol{w}_i - \boldsymbol{x}_i(k) \tag{7.39}$$

$$\boldsymbol{v}_i^{(2)} = \left[\boldsymbol{v}_{i(2)}^{(1)}, \boldsymbol{v}_{i(1)}^{(1)}\right] \tag{7.40}$$

$$\boldsymbol{v}_i^* = \boldsymbol{v}_i^{(2)} - \frac{\langle \boldsymbol{v}_i^{(2)}, \boldsymbol{v}_i^{(1)} \rangle}{\langle \boldsymbol{v}_i^{(1)}, \boldsymbol{v}_i^{(1)} \rangle} \boldsymbol{v}_i^1 \tag{7.41}$$

$$\widehat{\boldsymbol{u}}_i = \boldsymbol{v}_i^{(1)} + \boldsymbol{v}_i^* \tag{7.42}$$

6. **Steering (Random)**. In this control scheme, the vehicle i moves from waypoint to waypoint along a randomly curved path. We create this random path by forcing the vehicle to steer towards a randomly-generated intermediate waypoint between the vehicle and its current waypoint. When it reaches the intermediate waypoint, a new randomly-generated waypoint is created between the vehicle and its current waypoint on the path. This process is designed to cause the vehicle to eventually converge onto the current waypoint over time. To determine the desired velocity vector, we

must first calculate $\boldsymbol{v}_i^{(1)}$ and $\boldsymbol{v}_i^*$ the same way as in Equations 7.39 and 7.41, respectively. We must also generate two random variables: $\psi^{(1)} = \mathcal{U}(0,1)$ and $\psi^{(2)} = \mathcal{N}(0,1)$. From these values, we calculate the intermediate waypoint location $\boldsymbol{w}_i^*$, which we use to determine the desired velocity vector $\widehat{\boldsymbol{u}}_i$.

$$\boldsymbol{w}_i^* = \boldsymbol{x}_i(k) + \psi^{(1)}\boldsymbol{v}_i^{(1)} + \frac{\psi^{(1)}\psi^{(2)}\boldsymbol{v}_i^*}{2} \tag{7.43}$$

$$\widehat{\boldsymbol{u}}_i = \boldsymbol{w}_i^* - \boldsymbol{x}_i(k) \tag{7.44}$$

7. **Bad Vehicle**. In this control scheme, we simulate a vehicle i that behaves erratically with a probability of 5%. The erratic behavior comes about by switching the vehicle control from the “Always Moving” mode to the “Steering (Random)” mode for 20 seconds when a random variable from $\mathcal{U}(0,1)$ is found to be less than or equal to 0.05.

8. **Manual**. In this control scheme, any vehicle set to this mode is controlled by a human operator using the arrow keys on a computer keyboard. The spacebar acts as an emergency stop that immediately halts all manually controlled vehicles. Using the current velocity $\dot{\boldsymbol{x}}_i(k)$, the manually desired velocity vector can be calculated as follows.

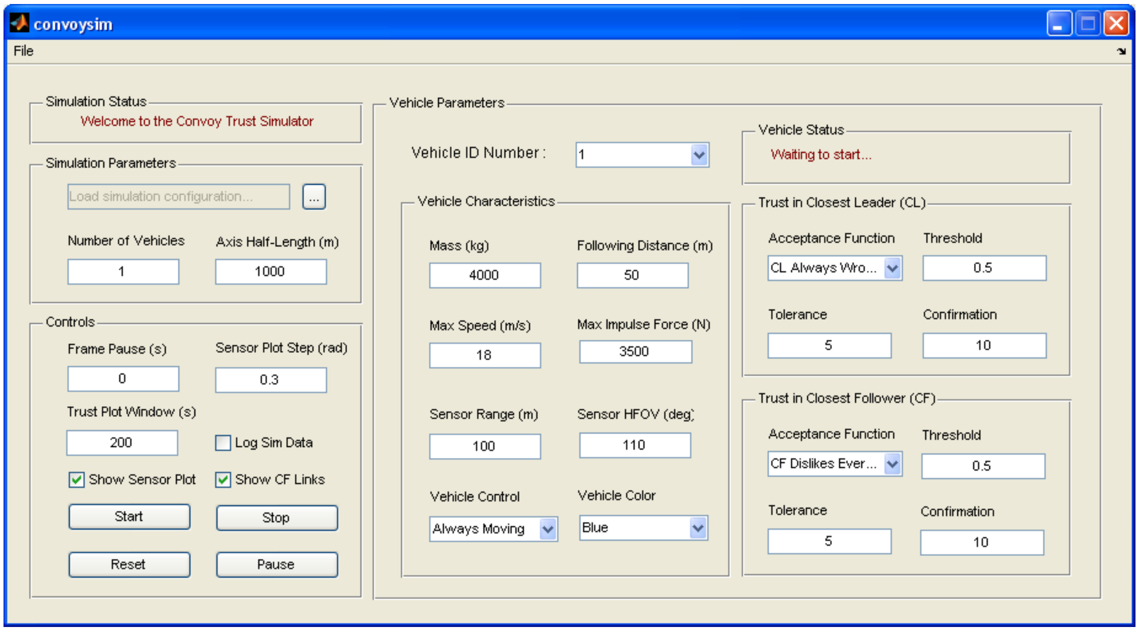
$$\widehat{\boldsymbol{u}}_i = \begin{cases} \dot{\boldsymbol{x}}_i(k) + (-0.25, 0) & \leftarrow \\ \dot{\boldsymbol{x}}_i(k) + (0.25, 0) & \rightarrow \\ \dot{\boldsymbol{x}}_i(k) + (0, 0.25) & \uparrow \\ \dot{\boldsymbol{x}}_i(k) + (0, -0.25) & \downarrow \\ \boldsymbol{0} & \text{spacebar} \end{cases} \tag{7.45}$$

7.2.3 Convoy Simulator (ConvoySim) Graphical User Interface (GUI)

The convoy simulator GUI is an application that runs on top of the Matlab programming environment (Figure 7.5). It provides a user with direct access to all of the simulation variables and options in a concise way to facilitate the rapid development of trust-based convoy simulation experiments. The GUI also validates every simulation, vehicle, or control parameter update to ensure that the user is unable to configure a simulation experiment in an invalid manner.

The GUI operates within two modes: configuration mode and execution mode. During configuration mode, a user is allowed to make valid modifications to all available parameters via the GUI interface in order to configure a simulation experiment. Configuration mode transitions to execution mode when the user presses the Start button to run the configured simulation experiment. This experiments runs until a user chooses to stop it by pressing the Stop button. While running, three plot figures are instantiated to provide the user with feedback about the experiment: Closest Leader Trust Plot, Closest Followers Trust Plot, and Convoy Trust Simulation Plot. Both trust plots display the trust value history within a particular time frame for the currently selected vehicle. The simulation plot provides an overhead real-time view of the world, where red crosses indicate waypoints and colored dots indicate vehicles.

The remainder of this section provides detailed descriptions of each simulation GUI parameter and control. We also provide Tables 7.1 through 7.5, which list these parameters and controls with respect to their property attributes, including its name, type, default value, range, and math variable.



Simulation Status
Welcome to the Convoy Trust Simulator

Simulation Parameters
Load simulation configuration...
Number of Vehicles: 1
Axis Half-Length (m): 1000

Controls
Frame Pause (s): 0
Sensor Plot Step (rad): 0.3
Trust Plot Window (s): 200
□ Log Sim Data
☑ Show Sensor Plot
☑ Show CF Links

Vehicle Parameters
Vehicle ID Number : 1

Vehicle Status
Waiting to start...

Vehicle Characteristics
Mass (kg): 4000
Following Distance (m): 50
Max Speed (m/s): 18
Max Impulse Force (N): 3500
Sensor Range (m): 100
Sensor HFOV (deg): 110
Vehicle Control: Always Moving
Vehicle Color: Blue

Trust in Closest Leader (CL)
Acceptance Function: CL Always Wro...
Threshold: 0.5
Tolerance: 5
Confirmation: 10

Trust in Closest Follower (CF)
Acceptance Function: CF Dislikes Ever...
Threshold: 0.5
Tolerance: 5
Confirmation: 10

Figure 7.5. A Screenshot of the Matlab ConvoySim Application. (This figure is presented in color; the black and white reproduction may not be an accurate representation.)

Table 7.1.

ConvoySim Simulation Parameters

Name	Type	Default Value	Units	Range	Math Variable
Load Sim. Config.	Button	N/A	N/A	N/A	N/A
Number of Vehicles	Textbox	1	N/A	$[1,\infty)$	$\|N\|$
Axis Half-Length	Textbox	1000	Meters	$[1,\infty)$	N/A

Table 7.2.

ConvoySim Controls

Name	Type	Default Value	Units	Range	Math Variable
Frame Pause	Textbox	0	N/A	$[0,\infty)$	N/A
Sensor Plot Step	Textbox	0.3	Radians	$(0,2\pi)$	N/A
Trust Plot Window	Textbox	200	Seconds	$[1,\infty)$	N/A
Log Sim Data	Checkbox	Unchecked	N/A	N/A	N/A
Show Sensor Plot	Checkbox	Checked	N/A	N/A	N/A
Show CF Links	Checkbox	Checked	N/A	N/A	N/A
Start	Button	N/A	N/A	N/A	N/A
Stop	Button	N/A	N/A	N/A	N/A
Reset	Button	N/A	N/A	N/A	N/A
Pause	Toggle Button	N/A	N/A	N/A	N/A

Table 7.3.

ConvoySim Vehicle Parameters

Name	Type	Default Value	Units	Range	Math Variable
Vehicle ID Number	List	1	N/A	$[1, \lvert N \rvert]$	$i \in N \subset \mathbb{N}^+$
Mass	Textbox	4000	Kilograms	$(0, \infty)$	$m_i \in \mathbb{R}$
Following Distance	Textbox	50	Meters	$(0, \infty)$	$d_i \in \mathbb{R}$
Max Speed	Textbox	18	Meters per Second	$(0, \infty)$	$s_i \in \mathbb{R}$
Max Impulse Force	Textbox	3500	Newtons per Second	$(0, \infty)$	$p_i \in \mathbb{R}$
Sensor Range	Textbox	100	Meters	$(0, \infty)$	$r_i \in \mathbb{R}$
Sensor HFOV	Textbox	110	Degrees	(0,360]	$\theta_i \in \mathbb{R}$
Vehicle Control	List	“Always Moving”	N/A	N/A	N/A
Vehicle Color	List	“Blue”	N/A	N/A	N/A

Table 7.4.

ConvoySim Trust in Closest Leader (CL)

Name	Type	Default Value	Units	Range	Math Variable
Acceptance Function	List	“CL Always Wrong”	N/A	N/A	N/A
Threshold	Textbox	0.5	N/A	[0,1]	$t_i^{(\mathcal{L})} \in \mathbb{R}$
Tolerance	Textbox	5	N/A	$[0, c_i^{(\mathcal{L})}]$	$\tau_i^{(\mathcal{L})} \in \mathbb{N}$
Confirmation	Textbox	10	N/A	$[\tau_i^{(\mathcal{L})}, \infty]$	$c_i^{(\mathcal{L})} \in \mathbb{N}$

Table 7.5.

ConvoySim Trust in Closest Follower (CF)

Name	Type	Default Value	Units	Range	Math Variable
Acceptance Function	List	“CF Dislikes Everything”	N/A	N/A	N/A
Threshold	Textbox	0.5	N/A	[0,1]	$t_i^{(\mathcal{F})} \in \mathbb{R}$
Tolerance	Textbox	5	N/A	$[0, c_i^{(\mathcal{F})}]$	$\tau_i^{(\mathcal{F})} \in \mathbb{N}$
Confirmation	Textbox	10	N/A	$[\tau_i^{(\mathcal{F})}, \infty]$	$c_i^{(\mathcal{F})} \in \mathbb{N}$

The **“Simulation Parameters”** panel defines a group of controls used to configure the global parameters for a simulation experiment.

- **Number of Vehicles**. Indicates the total quantity of vehicles in the simulation. This parameter also updates the length of the dropdown list for Vehicle ID Number.
- **Axis Half-Length**. This parameter is set to scale the size of the world. It indicates the half-length (in meters) of each axis in the xy plane. This parameter also supports the absolute positioning of waypoints in the world, since waypoints in the simulation data file are provided relative to an xy plane where values are bounded by $-1 \leq x \leq 1$ and $-1 \leq y \leq 1$.
- **Load Simulation Configuration**. This button opens a dialog box that allows a user to select a simulation data file from a previously saved simulation and restore all GUI parameters to the same state as the saved

simulation. Waypoints within the simulation data file are configured externally to the GUI.

The **"Controls"** panel provides both interactive capabilities and visualization preferences during the simulation execution mode.

- **Frame Pause**. This parameter indicates the amount of time to pause (in seconds) between the rendering of each simulation frame. This is useful to slow down the execution of the simulation on fast computers so that the user might be able to comprehend the simulation visualization.
- **Sensor Plot Step**. This parameter affects the visual curvature of the sensor sector for each vehicle. In general, larger values result in less curvy sectors. But larger values also speed up the execution time between each frame.
- **Trust Plot Window**. This parameter sets the maximum number of data points to plot within each trust plot figure. In general, larger values allow a user to see more historical data about the trust dynamics at once. But larger values also slow down the execution time between each frame.
- **Log Sim Data**. This checkbox, when checked, triggers the opening of a dialog box at the end of a simulation to allow a user to save the results of a simulation run into a Matlab data file. The default file name is given as "ConvoyTrustDataLog.mat".
- **Show Sensor Plot**. This checkbox, when checked, renders the sensor sector for each vehicle in the Convoy Trust Simulation figure. The sensor sector visualization not only indicates the forward direction of each vehicle; it also indicates the quality of the observation of the closest leader in range. Blue,

red, and green sector colors indicate no observation, unfavorable observation, and favorable observation, respectively. The shape of the sensor sector for each vehicle is dictated by the Sensor Range and Sensor HFOV vehicle parameters.

- **Show CF Links**. This checkbox, when checked, renders a line between two vehicles to indicate the quality of the observation of the closest followers in range. A red line indicates an unfavorable observation and a green line indicates a favorable observation.
- **Start Button**. This button, when pressed, starts the convoy simulation that is configured within the GUI. All GUI controls, except for the Stop button and Vehicle ID Number dropdown list, are disabled while a simulation is being executed.
- **Stop Button**. This button, when pressed, stops the execution of the convoy simulation.
- **Reset Button**. This button, when pressed, resets all GUI parameters to their default values.
- **Pause Button**. This toggle button pauses the execution of the simulation.

The **"Vehicle Parameters"** panel defines the parameters for each individual vehicle, which is selected through the Vehicle ID Number dropdown list. This panel contains three sub-panels: Vehicle Characteristics, Trust in Closest Leader (CL), and Trust in Closest Follower (CF).

- **Mass**. This parameter defines the point mass of the vehicle in kilograms.
- **Following Distance**. This parameter defines the minimum following distance (in meters) that a vehicle will attempt to maintain while following a leader.
- **Max Speed**. This parameter defines the maximum speed (in meters per second) that a vehicle is allowed to travel in the simulation.
- **Max Impulse Force**. This parameter defines the maximum magnitude of the impulse force (in Newtons) that a vehicle is allowed to exert in any direction.
- **Sensor Range**. This parameter defines the maximum distance (in meters) that a vehicle can sense other vehicles within its proximity.
- **Sensor HFOV**. This parameter defines the horizontal field of view of the vehicle sensor (in degrees), which defines the sector of the sensor.
- **Vehicle Control**. This parameter selects a default vehicle control scheme from the possible control scheme listed in Section 7.3.2.3.
- **Vehicle Color**. This parameter assigns a vehicle color from the Matlab VGA colormap.
- **Acceptance Function**. This parameter selects the desired acceptance function to interpret observations gathered from the sensor data or communication messages. Acceptance functions to evaluate the closest leaders and followers are defined in Sections 7.3.2.1 and 7.3.2.2, respectively.

- **Threshold**. This parameter sets the desired trust threshold for a particular context. Trust values for vehicles that are greater than the trust threshold indicate that those vehicles are considered to be trustworthy; otherwise, those vehicles are considered to be untrustworthy.
- **Tolerance**. This parameter sets the desired tolerance for the RoboTrust model. Higher values indicate higher tolerance for unacceptable observations, suggesting a slower trust decay in response to unacceptable observations.
- **Confirmation**. This parameter sets the desired confirmation parameter for the RoboTrust model. High values indicate a need for more confirmation for acceptable observations, which suggests a slower trust growth in response to acceptable observations.

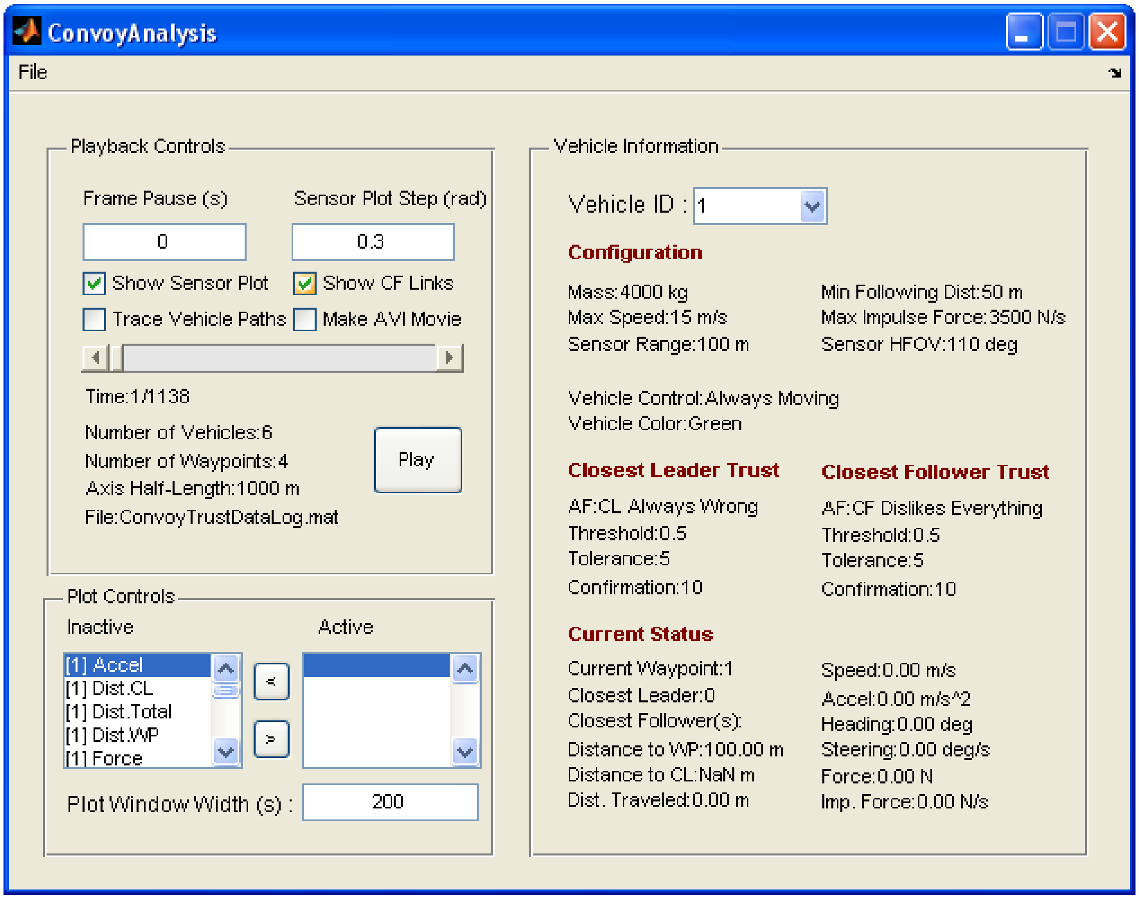
The GUI also includes two status panels as a way to provide text-based feedback to the user. The **"Simulation Status"** panel provides feedback regarding committed changes during simulation configuration and the current time during simulation execution. The **"Vehicle Status"** panel provides feedback about a specific vehicle during simulation execution, including its current waypoint, closest leader, closest followers, and speed.

7.3 Trust-Based Convoy Simulation Case Studies

In this section, we analyze and discuss three selected case studies for our trust-based vehicle controller. The first case analyzes how the trust-based controller switches between its default control mode and leader-following mode. The second case analyzes how a leader switches between its default control mode and leading-follower mode with respect to different contexts. And the third case demonstrates how the trust-based controller can detect and mitigate the bad behavior of another vehicle in the convoy. All case studies in this section were analyzed using the custom log analyzing tool, ConvoyAnalysis.

7.3.1 Convoy Log Analysis Tool (ConvoyAnalysis)

The convoy log analysis tool, named ConvoyAnalysis, is a GUI application that runs on top of the Matlab programming environment (Figure 7.6). It provides the user with a way to parse through a saved simulation data file in order to extract meaningful information from the simulation run. Upon loading a data file into memory, the tool also automatically generates derived time series data, such as speed and acceleration time series from vehicle position time series, in order facilitate deep data analysis. All data in memory is then linked to specific plot figures and visually displayed at the appropriate time frame, given a user's visualization preferences. There are four plot figures generated by the analysis tool: the Active Time Series plot, Trust in Closest Leaders plot, Trust in Closest Followers plot, and Convoy Trust Simulation plot. The ConvoyAnalysis GUI is divided into three panels: Vehicle Information, Plot Controls, and Playback Controls. The Vehicle Information panel is dedicated to presenting the user with vehicle information and has one control: a dropdown list of vehicle identification numbers. Upon selecting a particular vehicle number, the text data in the panel updates to reflect the selected vehicle's configuration, trust preferences for leaders and followers, and its current status at a particular time. Also, the trust plot



Playback Controls

Frame Pause (s): 0

Sensor Plot Step (rad): 0.3

☑ Show Sensor Plot ☑ Show CF Links

□ Trace Vehicle Paths □ Make AVI Movie

Time:1/1138

Number of Vehicles:6
Number of Waypoints:4
Axis Half-Length:1000 m
File:ConvoyTrustDataLog.mat

Plot Controls

Inactive
[1] Accel
[1] Dist.CL
[1] Dist.Total
[1] Dist.WP
[1] Force

Active

Plot Window Width (s) : 200

Vehicle Information

Vehicle ID : 1

Configuration

Mass:4000 kg
Max Speed:15 m/s
Sensor Range:100 m

Min Following Dist:50 m
Max Impulse Force:3500 N/s
Sensor HFOV:110 deg

Vehicle Control:Always Moving
Vehicle Color:Green

Closest Leader Trust

AF:CL Always Wrong
Threshold:0.5
Tolerance:5
Confirmation:10

Closest Follower Trust

AF:CF Dislikes Everything
Threshold:0.5
Tolerance:5
Confirmation:10

Current Status

Current Waypoint:1
Closest Leader:0
Closest Follower(s):
Distance to WP:100.00 m
Distance to CL:NaN m
Dist. Traveled:0.00 m

Speed:0.00 m/s
Accel:0.00 m/s^2
Heading:0.00 deg
Steering:0.00 deg/s
Force:0.00 N
Imp. Force:0.00 N/s

Figure 7.6. A Screenshot of the Matlab ConvoyAnalysis Application. (This figure is presented in color; the black and white reproduction may not be an accurate representation.)

figures update to show the trust value histories for the selected vehicle's leaders and followers. The Plot Controls panel is used to configure the display of the Active Time Series and trust plots. The Plot Window Width text box sets the maximum number of data points to plot in the three plots. The Active list indicates which time series will be displayed in the Active Time Series plot, and is populated by the choices in the Inactive list through the use of the arrow buttons between the lists. The Playback Controls panel primarily contains controls that affect the current time in a simulation. These controls are the horizontal time slider bar and the Play/Stop toggle button, which allow a user to select a specific current time and play back the saved simulation from the selected time. The other controls in panel influence the display of the vehicles in the Convoy Trust Simulation plot.

7.3.2 Case Study 1: Stop and Go

The purpose of our first case study is to demonstrate the trust-based controller switching between the default control mode and leader following mode. We set up the scenario with two vehicles, configured according to Table 7.6, using the default square waypoint configuration.

$$W = \{(500,500), (500,-500), (-500,-500), (-500,500)\} \tag{7.46}$$

The variable in this case study is the Leader Trust Threshold for vehicle 2, $t_2^{(\mathcal{L})}$. Our study will vary $t_2^{(\mathcal{L})}$ between 0.45 and 0.5, and observe the change in vehicle 2's behavior, particularly in the moments before and after a mode switch. Vehicle 2's context for evaluating the trustworthiness in vehicle 1 only considers whether or not vehicle 1 is moving. As such, because vehicle 1 has its default control set to stop for 10

Table 7.6.

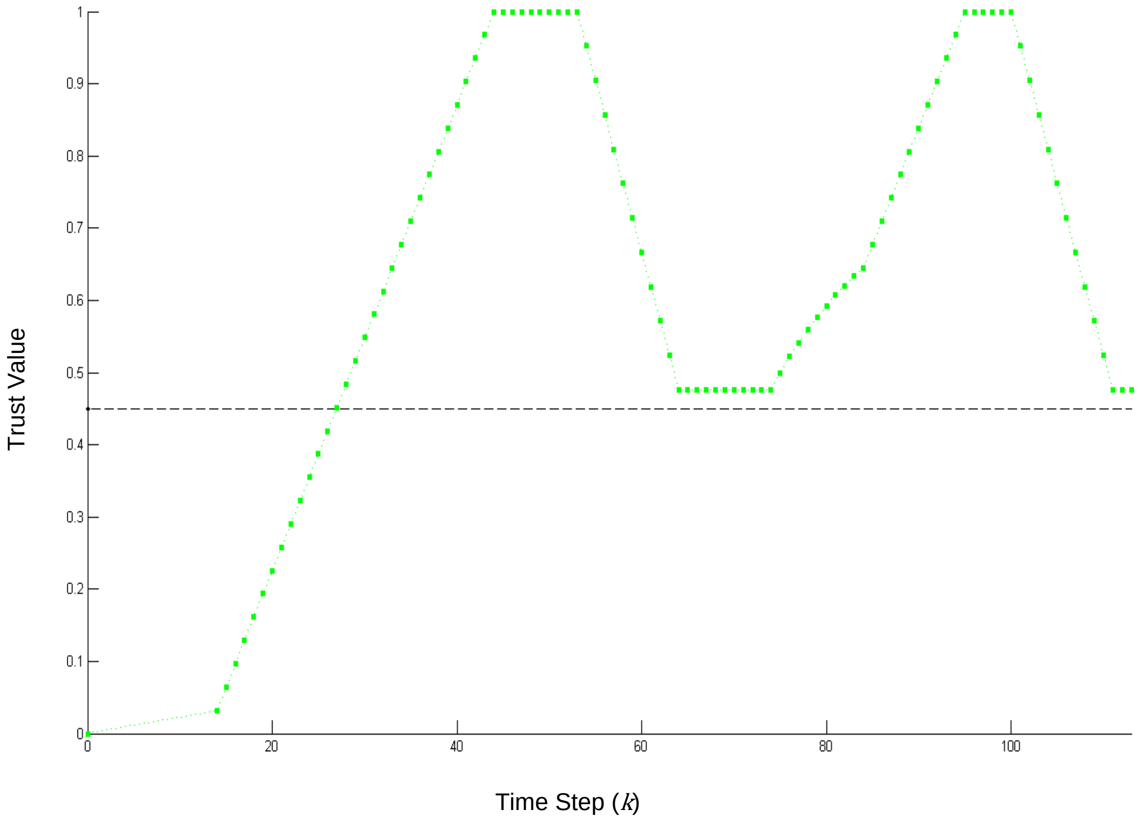
Case Study 1 Vehicle Configuration

Parameter	**Vehicle 1**	**Vehicle 2**
Mass	4000 kg	4000 kg
Min Following Distance	50 m	**75 m**
Max Speed	**15 m/s**	20 m/s
Max Impulse Force	3500 N/s	3500 N/s
Sensor Range	100 m	100 m
Sensor HFOV	110°	110°
Control Scheme	**Stop (10s) & Go**	Always Moving
Color	Green	Blue
Leader Acceptance Function	CL Always Wrong	**CL Is Moving**
Leader Trust Threshold	0.5	**{0.45, 0.5}**
Leader Tolerance	5	20
Leader Confirmation	10	30
Follower Acceptance Function	CF Dislikes Everything	CF Dislikes Everything
Follower Trust Threshold	0.5	5
Follower Tolerance	5	5
Follower Confirmation	10	10

seconds after it reaches its maximum speed, we can expect that vehicle 2 will lose trust in vehicle 1 during the moment vehicle 1's speed is equal to zero.

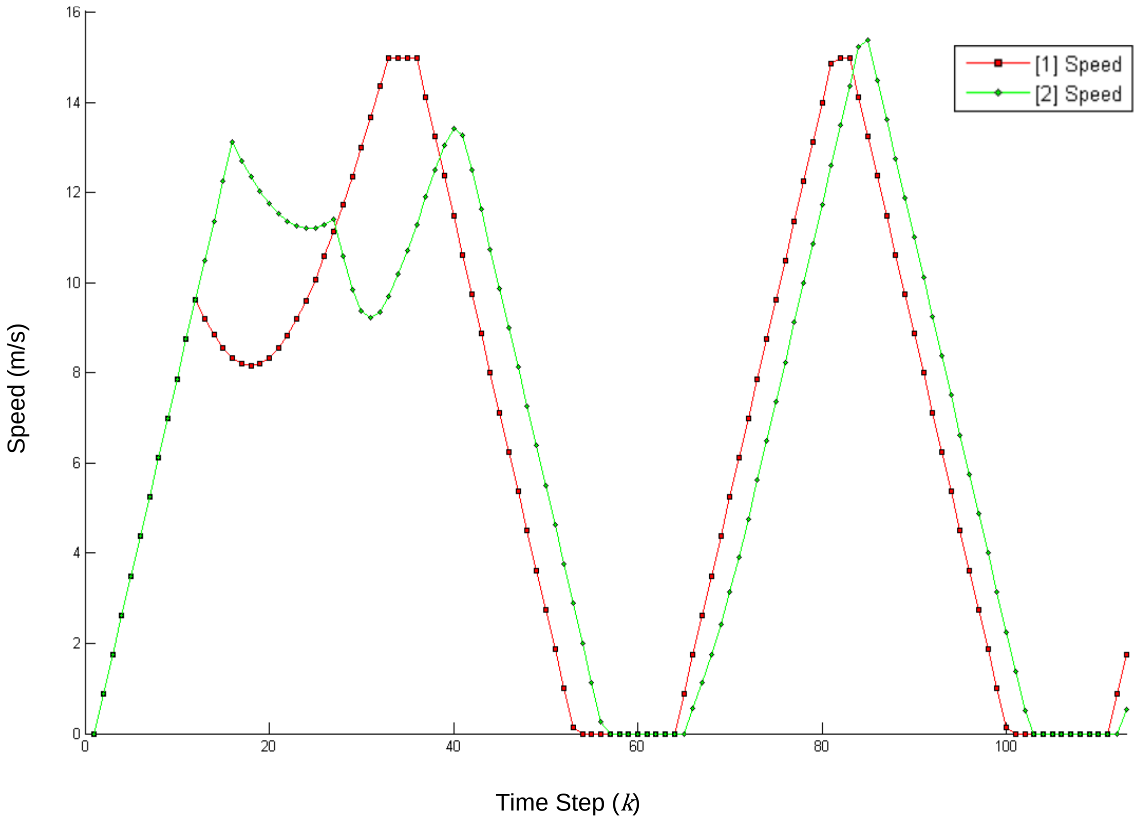
We present the simulation results when $t_2^{(\mathcal{L})} = 0.45$ in Figure 7.7. Figure 7.7a shows us the time series plot of vehicle 2's trust toward vehicle 1 with respect to the context that it is moving. Initial observations of vehicle 1 show favorable results, and at $k = 28$, vehicle 2 switches to leader following mode when $\boldsymbol{T}_{21}^{(\mathcal{L})} > 0.45$. At this point, Figure 7.7b shows vehicle 2 reducing its speed since it is too close to vehicle 1 – only 47 m away (Figure 7.7c), which is less than the set minimum following distance of 75 m. At $k = 37$, however, vehicle 1 begins to brake and reaches a complete stop at $k = 54$. It is at this point when vehicle 2 logs its first unfavorable observation of vehicle 1, resulting in a lower trust value toward vehicle 1. During the next 10 seconds, while vehicle 1 is stationary, vehicle 2's trust toward vehicle 1 continues to decline. However, at $k = 65$, vehicle 1 begins to move again, causing vehicle 2 to log a favorable observation. This favorable observation stabilizes the trust value, and because this value is still higher than 0.45, vehicle 2 remains in leader following mode. At $k = 66$, vehicle 2 begins to move as well, although at a slightly slower speed in order to increase its following distance, which was at 46.54 m. At $k = 84$, vehicle 1 brakes again and reaches a complete stop at $k = 101$. Again, we see vehicle 2 repeating the same trust dynamics as it had at $k = 54$.

In Figure 7.8, we show the simulation results when $t_2^{(\mathcal{L})} = 0.5$. The results mirror the previous results until $k = 64$. It is at this point when vehicle 2's trust toward vehicle 1 falls below the threshold (Figure 7.8a), causing vehicle 2 to switch back to its



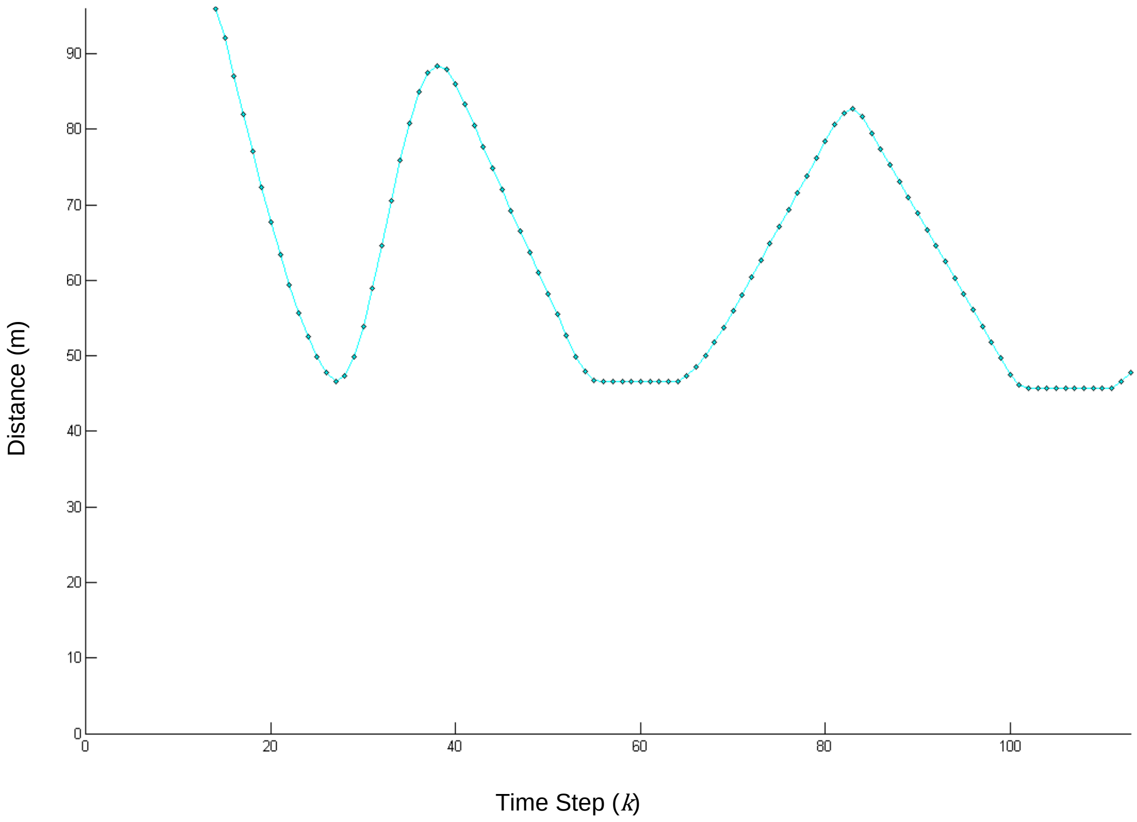
(a)

Figure 7.7. Simulation Results of Case Study 1 when $t_2^{(\mathcal{L})} = 0.45$. The plot in (a) shows vehicle 2's trust value towards vehicle 1 with respect to time. The plot in (b) shows the speed of both vehicles with respect to time. The plot in (c) shows the distance between both vehicles with respect to time. (This figure is presented in color; the black and white reproduction may not be an accurate representation.)



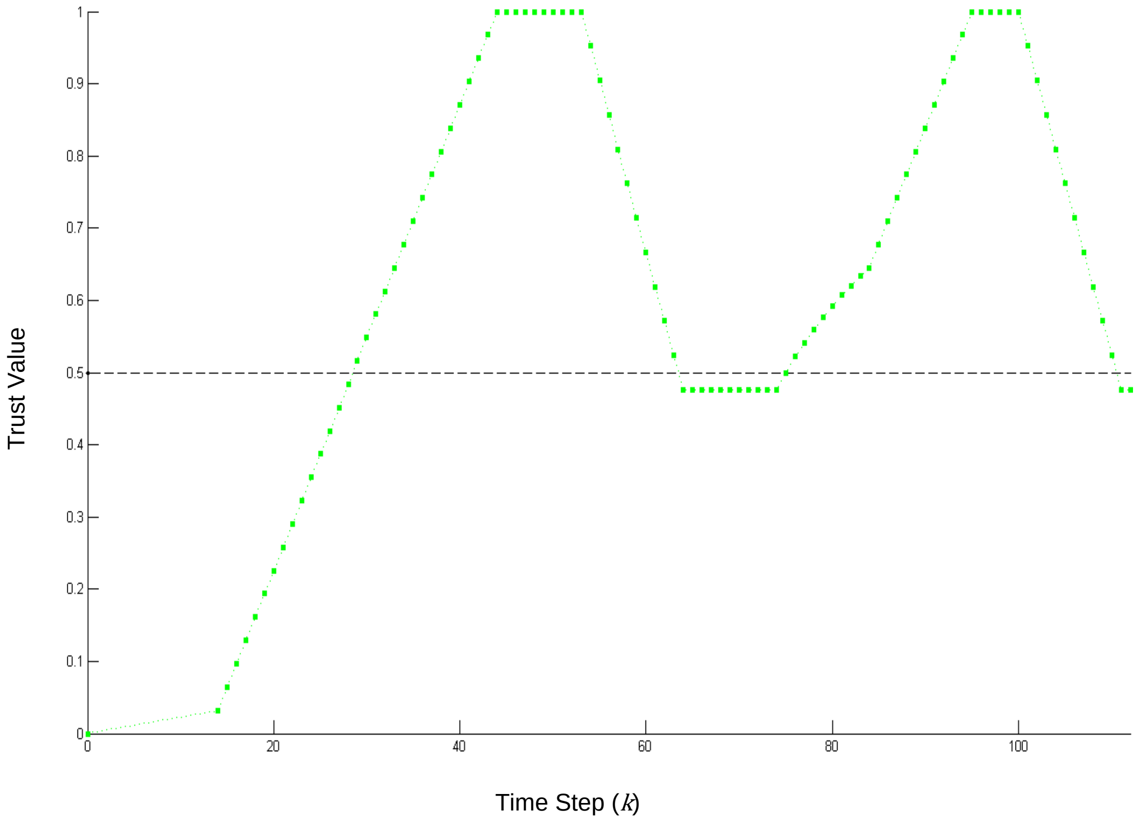
(b)

Figure 7.7 – Continued



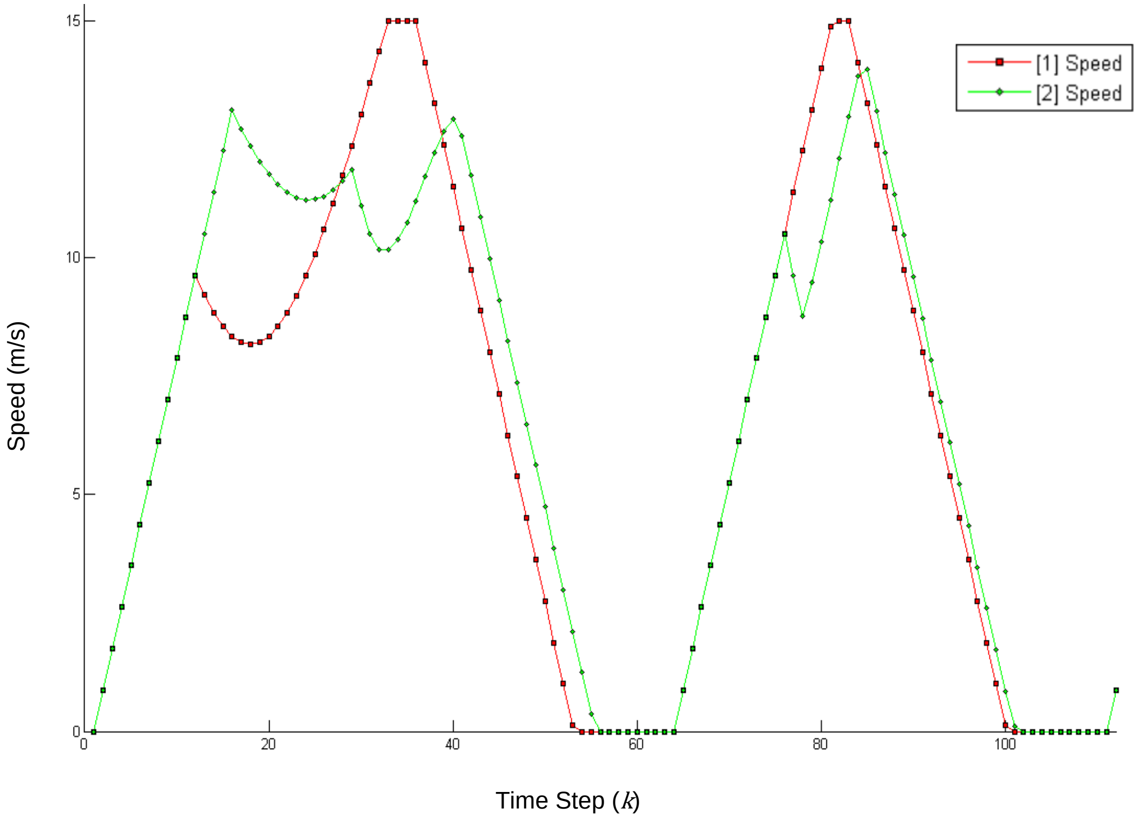
(c)

Figure 7.7 – Continued



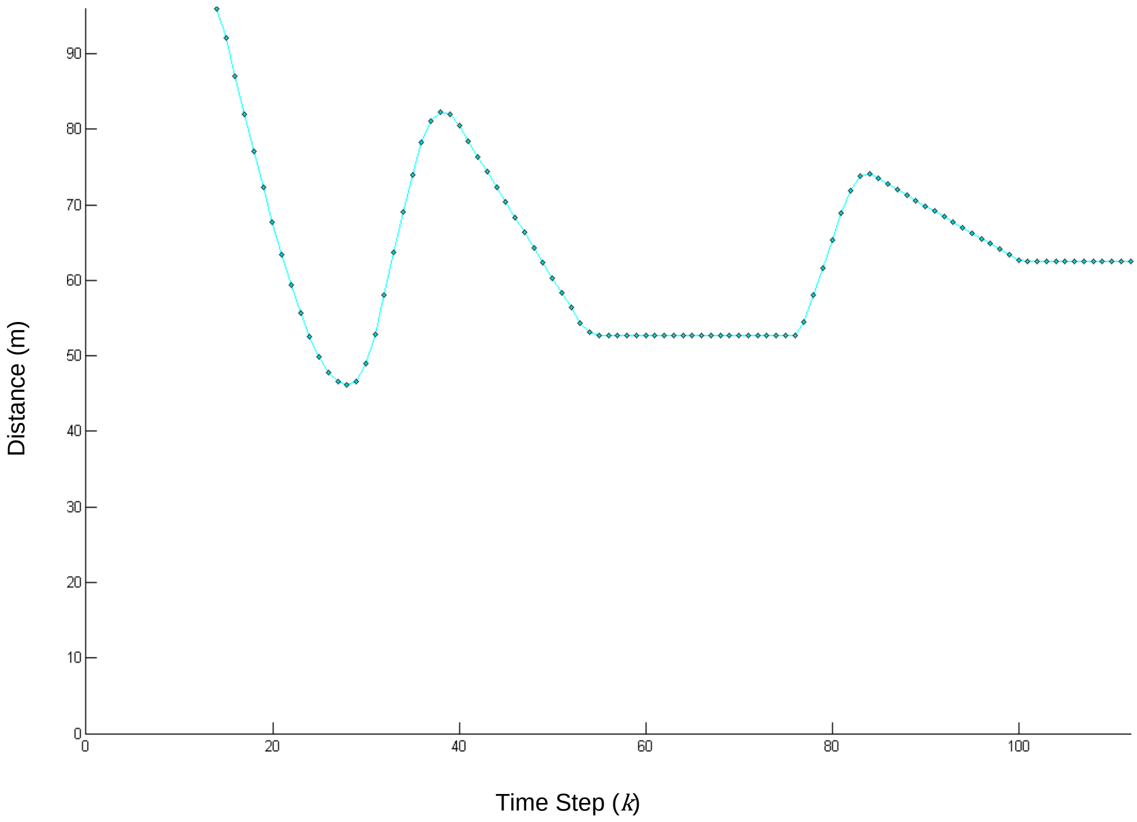
(a)

Figure 7.8. Simulation Results of Case Study 1 when $t_2^{(\mathcal{L})} = 0.5$. The plot in (a) shows vehicle 2's trust value towards vehicle 1 with respect to time. The plot in (b) shows the speed of both vehicles with respect to time. The plot in (c) shows the distance between both vehicles with respect to time. (This figure is presented in color; the black and white reproduction may not be an accurate representation.)



(b)

Figure 7.8 – Continued



(c)

Figure 7.8 – Continued

default control mode. Both vehicle 1 and 2 are shown to accelerate at the exact same time at $k = 65$ (Figure 7.8b). This said, vehicle 2 is not following vehicle 1 from a controls perspective. This can be observed in Figure 7.8c by noting that vehicle 2 makes no attempt to adjust its following distance to vehicle 1, such that it is less than its set minimum following distance of 75 m. Vehicle 2, however, is monitoring vehicle 1 since vehicle 1 remains within its sensor sector.   Vehicle 2 stays in its default control mode for the next 11 seconds, until its trust value toward vehicle 1 exceeds the threshold at $k = 76$. At this point, vehicle 2 returns to leader following mode, until its trust toward vehicle 1 once again falls below the threshold at $k = 111$, causing it to switch back to its default control mode.

7.3.3 Case Study 2: Stop at Waypoint

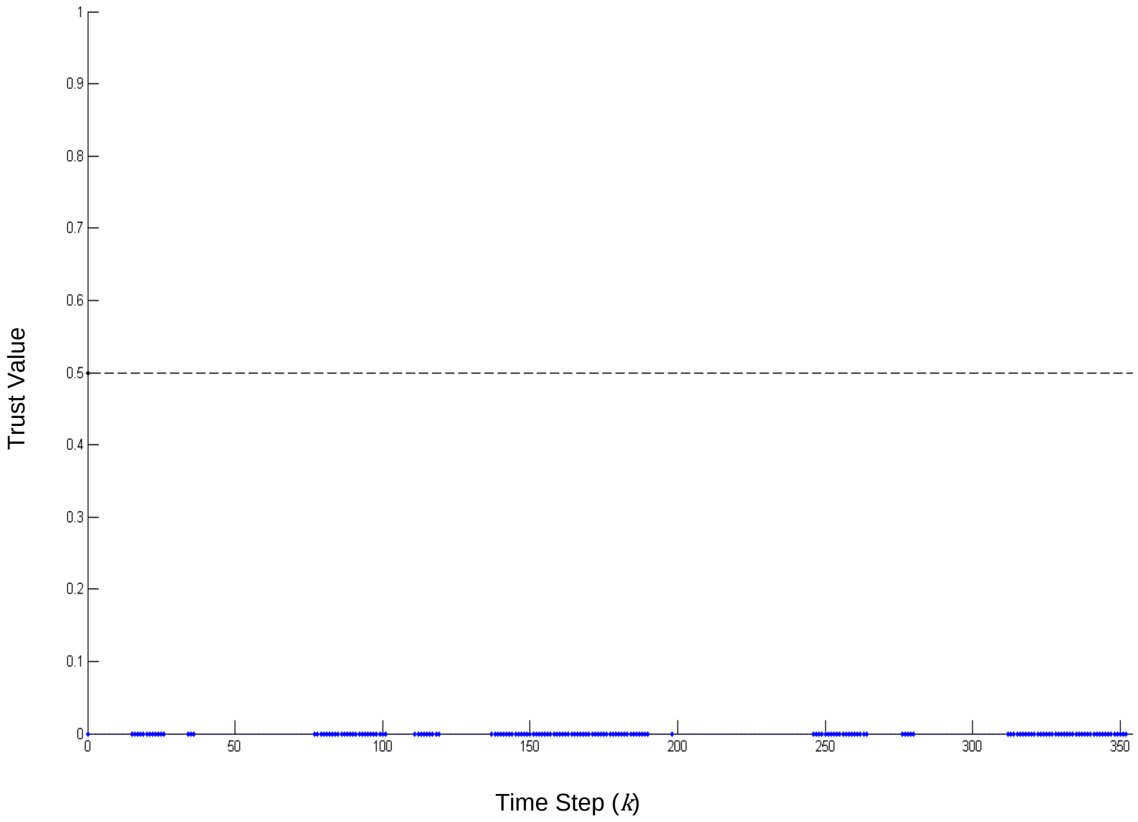
The purpose of our second case study is to demonstrate the trust-based controller adjusting a leader's trajectory to accommodate the perceived desires of a trusted follower. We set up the scenario with two vehicles, configured according to Table 7.7, using the default square waypoint configuration in Equation 7.46. The variable in this case study is the Follower Acceptance Function for vehicle 1. Our study will vary this acceptance function, leaving all other parameters fixed in both vehicles, and observe the behavioral changes in both vehicles. For brevity and comprehensibility, we will refer to vehicle 1 as the leader and vehicle 2 as the follower.

In Figure 7.9, we present our simulation results for the baseline case where the leader's Follower Acceptance Function is set to "CF Dislikes Everything." Essentially, this means that the leader will never adjust its trajectory in response to any

Table 7.7

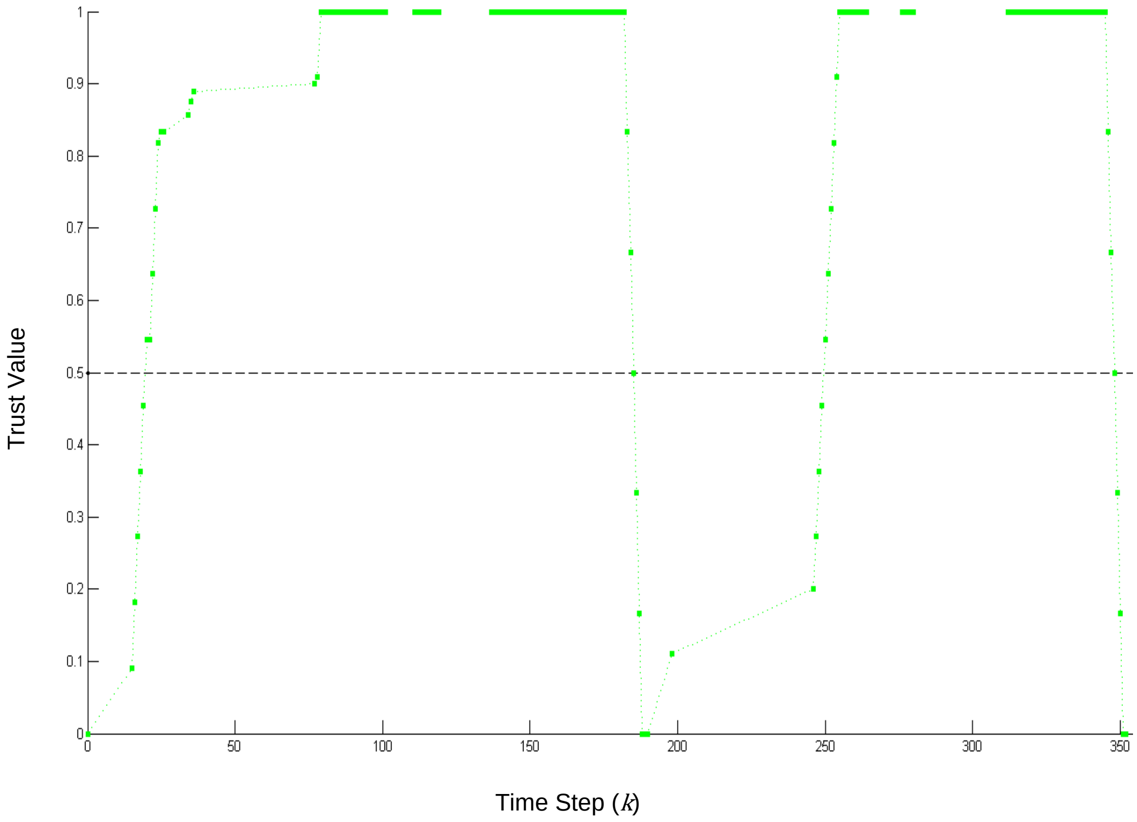
Case Study 2 Vehicle Configuration

Parameter	**Vehicle 1**	**Vehicle 2**
Mass	4000 kg	4000 kg
Min Following Distance	50 m	50 m
Max Speed	15 m/s	18 m/s
Max Impulse Force	4000 N/s	3600 N/s
Sensor Range	100 m	100 m
Sensor HFOV	110°	110°
Control Scheme	Stop at Waypoint	Stop at Waypoint
Color	Green	Blue
Leader Acceptance Function	CL Always Wrong	**CL Going To My WP**
Leader Trust Threshold	0.5	0.5
Leader Tolerance	5	5
Leader Confirmation	10	10
Follower Acceptance Function	{**CF Dislikes Everything, CF Likes My Waypoint, CF Likes My Heading**}	CF Dislikes Everything
Follower Trust Threshold	0.5	0.5
Follower Tolerance	15	5
Follower Confirmation	20	10



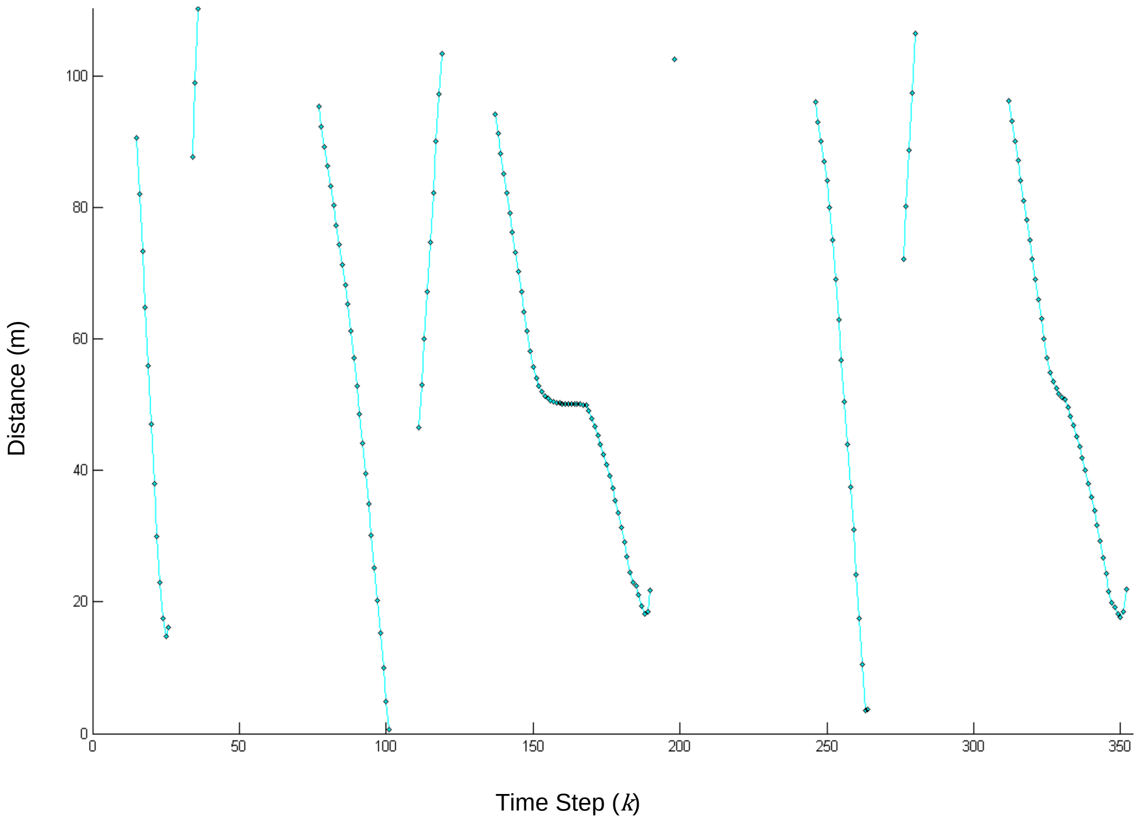
(a)

Figure 7.9. Simulation Results of Case Study 2 when Vehicle 1's Follower Acceptance Function is "CF Dislikes Everything." The plot in (a) shows vehicle 1's trust towards vehicle 2 with respect to time. The plot in (b) shows vehicle 2's trust towards vehicle 1 with respect to time. The plot in (c) shows the distance between both vehicles with respect to time. The plot in (d) shows the distance between vehicle 2 and its current waypoint with respect to time. (This figure is presented in color; the black and white reproduction may not be an accurate representation.)



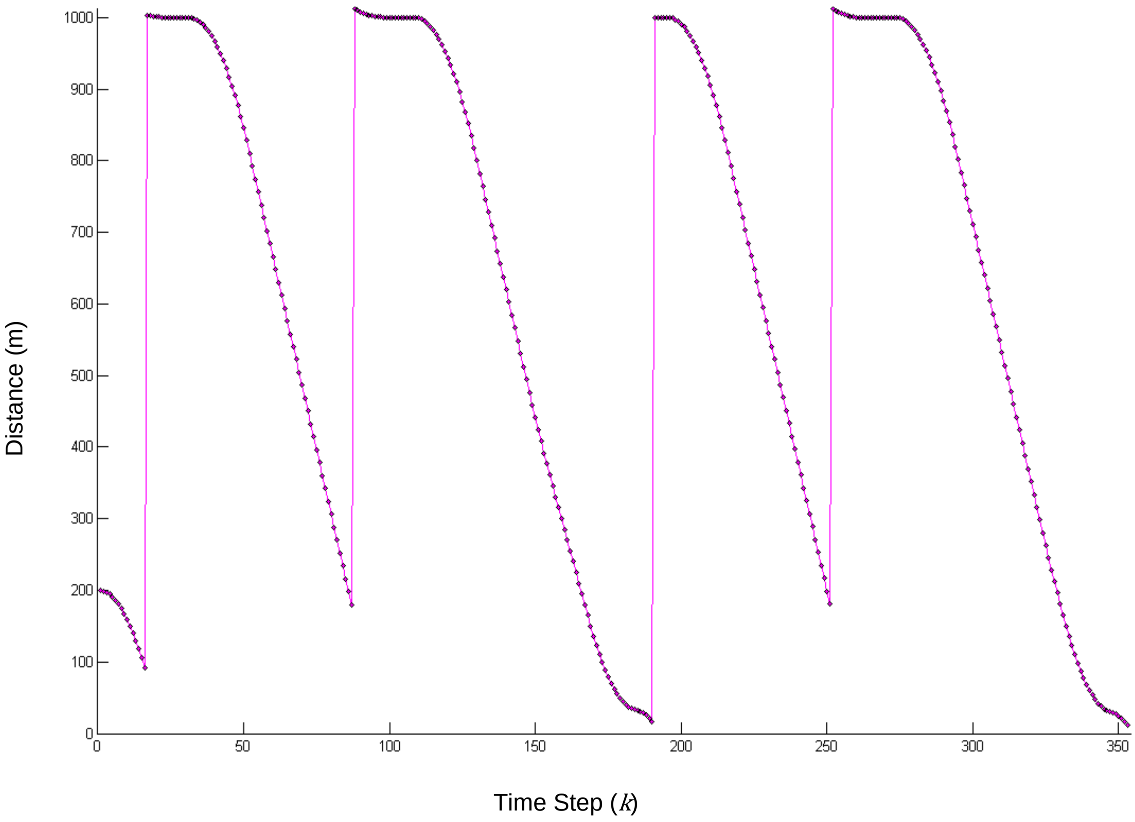
(b)

Figure 7.9 – Continued



(c)

Figure 7.9 – Continued

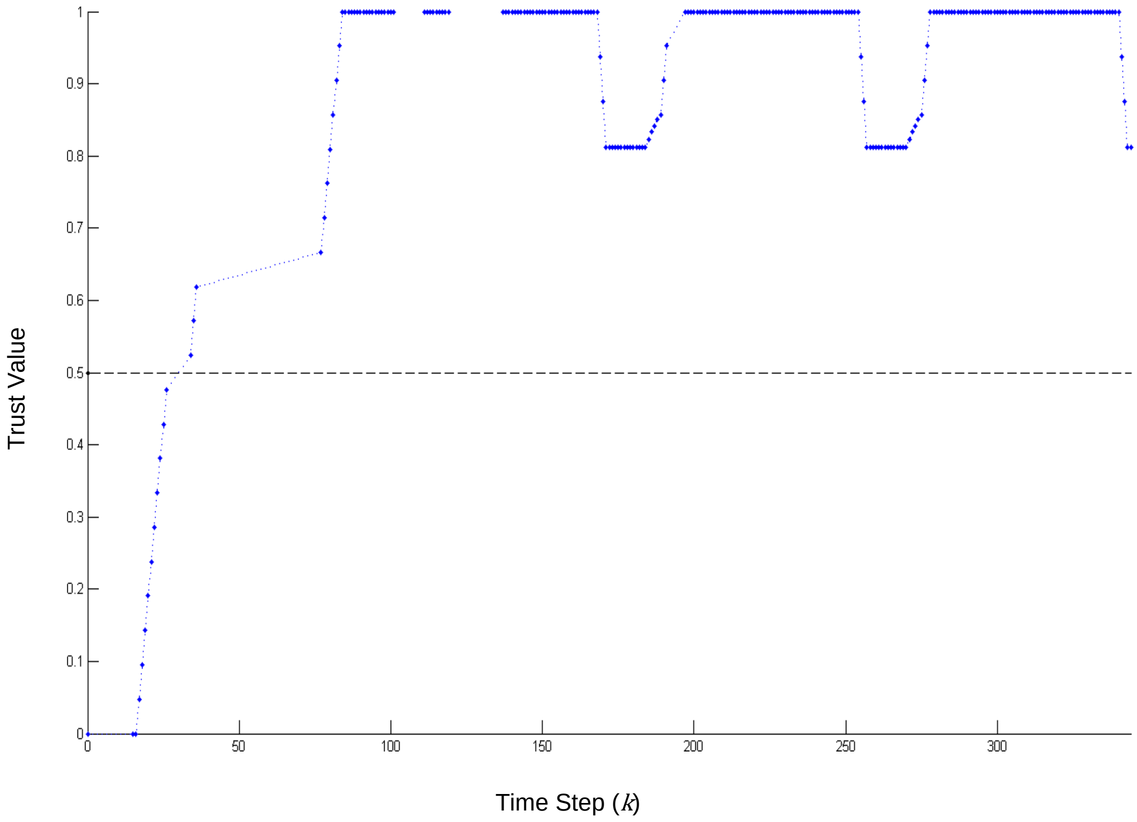


(d)

Figure 7.9 – Continued

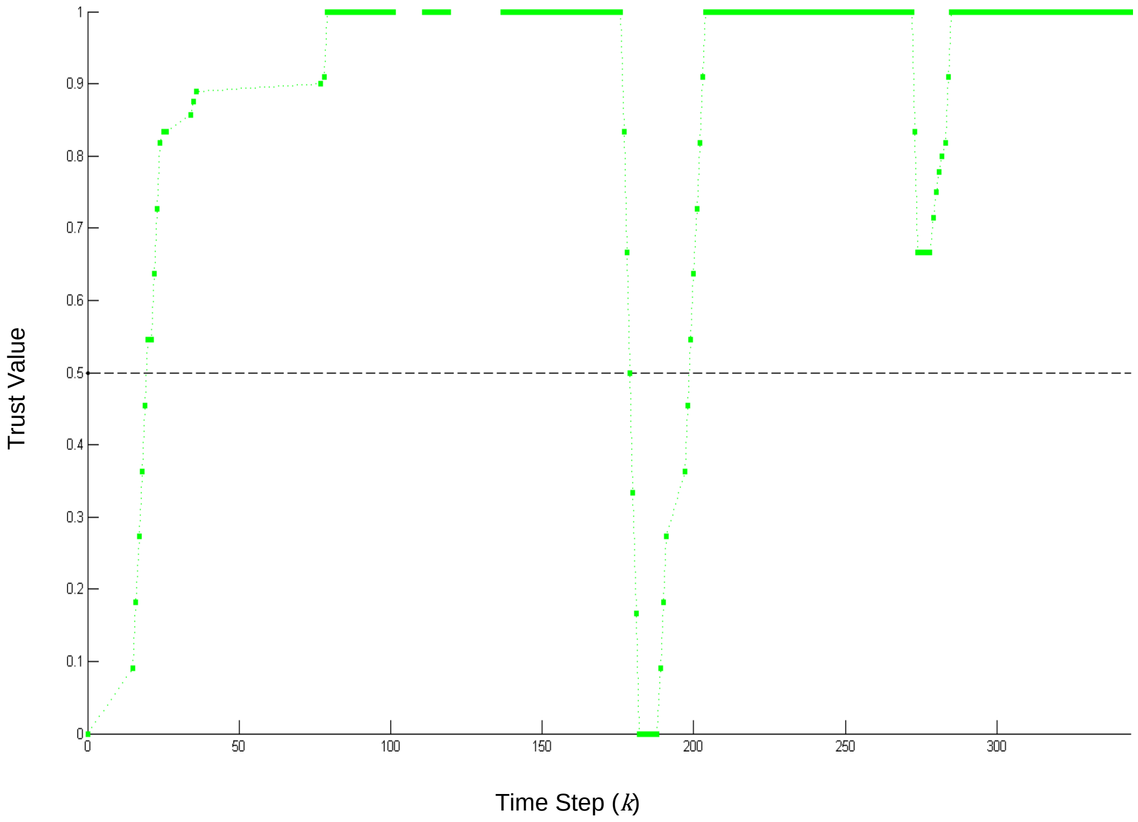
communication from the follower. In the baseline, the follower travelled a total of 4231.56 meters in 354 seconds within one revolution of the path, giving it an average speed of 11.95 meters per second. However, due to the disjoint nature of the observations and following distances plot in Figures 7.9a, 7.9b, and 7.9c, we can see that the leader and follower did not travel together for significant portions of the path. We can also see in Figure 7.9d that the follower's precision in triggering a waypoint switch was not precise. In some instances, a waypoint switch was triggered approximately 180 meters away from a waypoint; in other instances, the waypoint switch occurred as close as 16 meters away from a waypoint.

Figure 7.10 shows remarkably different results when the leader's Follower Acceptance Function is set to "CF Likes My WP." In this case, the leader only adjusts its trajectory if the trusted follower's waypoint is not the same as the leader's waypoint. Otherwise, the leader continues to proceed through the path using its default control mode. The results show that the follower travelled a total of 4170.59 meters in 344 seconds within one revolution of the path, giving it an average speed of 12.12 meters per second – an improvement over the baseline case. In addition, we see that there is more "togetherness" between the leader and the follower, as exhibited by more continuity in Figures 7.10a, 7.10b, and 7.10c. We can see why this happens by analyzing the transition from waypoint $W_{(3)}$ to $W_{(4)}$ for both vehicles. At $k = 169$, when the leader's current waypoint changes from $W_{(3)}$ to $W_{(4)}$, the leader logs its first unfavorable observation of the follower since the follower's current waypoint is $W_{(3)}$. But because the follower is trusted by the leader, the leader adjusts its trajectory to point



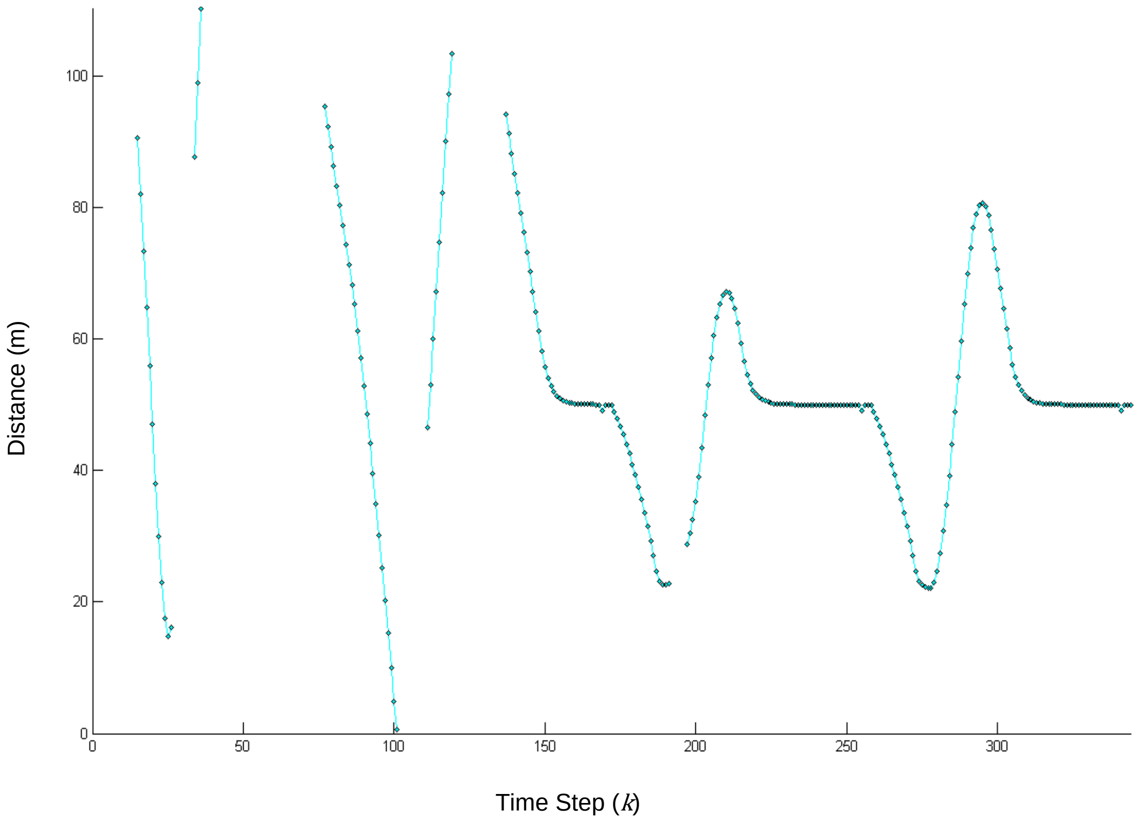
(a)

Figure 7.10. Simulation Results of Case Study 2 when Vehicle 1's Follower Acceptance Function is "CF Likes My Waypoint." The plot in (a) shows vehicle 1's trust towards vehicle 2 with respect to time. The plot in (b) shows vehicle 2's trust towards vehicle 1 with respect to time. The plot in (c) shows the distance between both vehicles with respect to time. The plot in (d) shows the distance between vehicle 2 and its current waypoint with respect to time. (This figure is presented in color; the black and white reproduction may not be an accurate representation.)



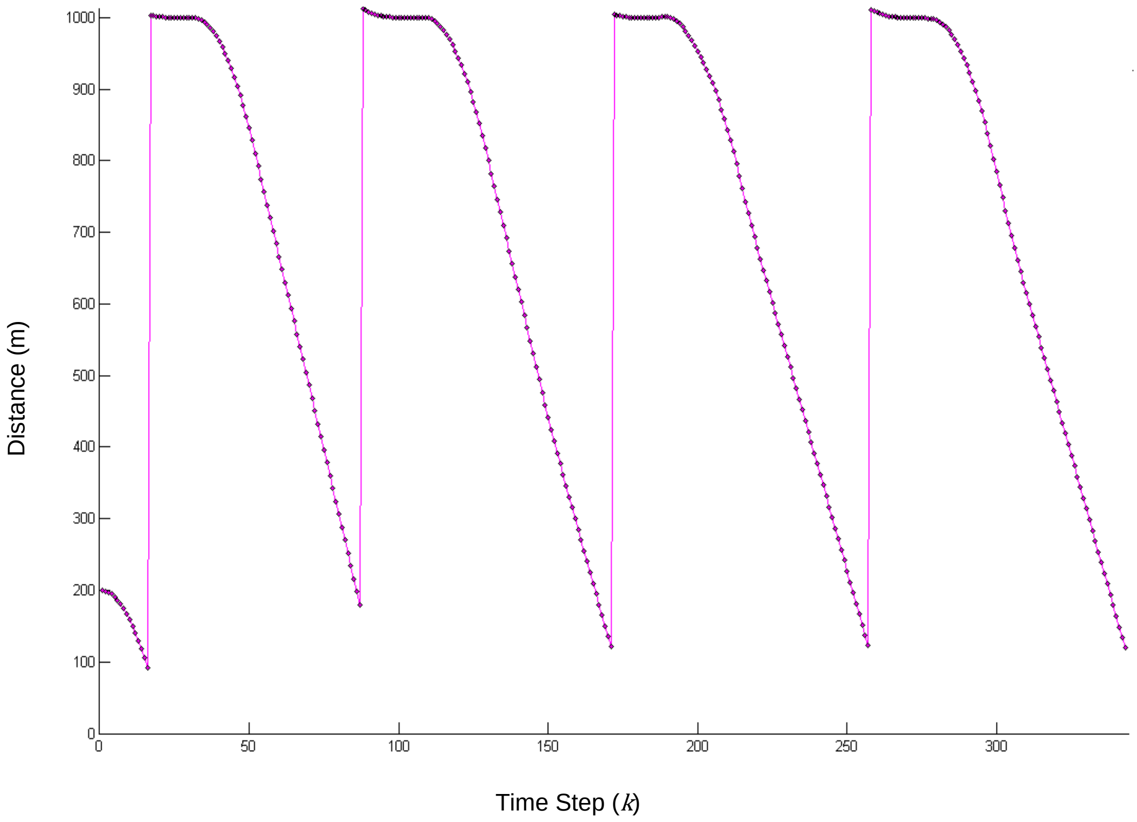
(b)

Figure 7.10 – Continued



(c)

Figure 7.10 – Continued



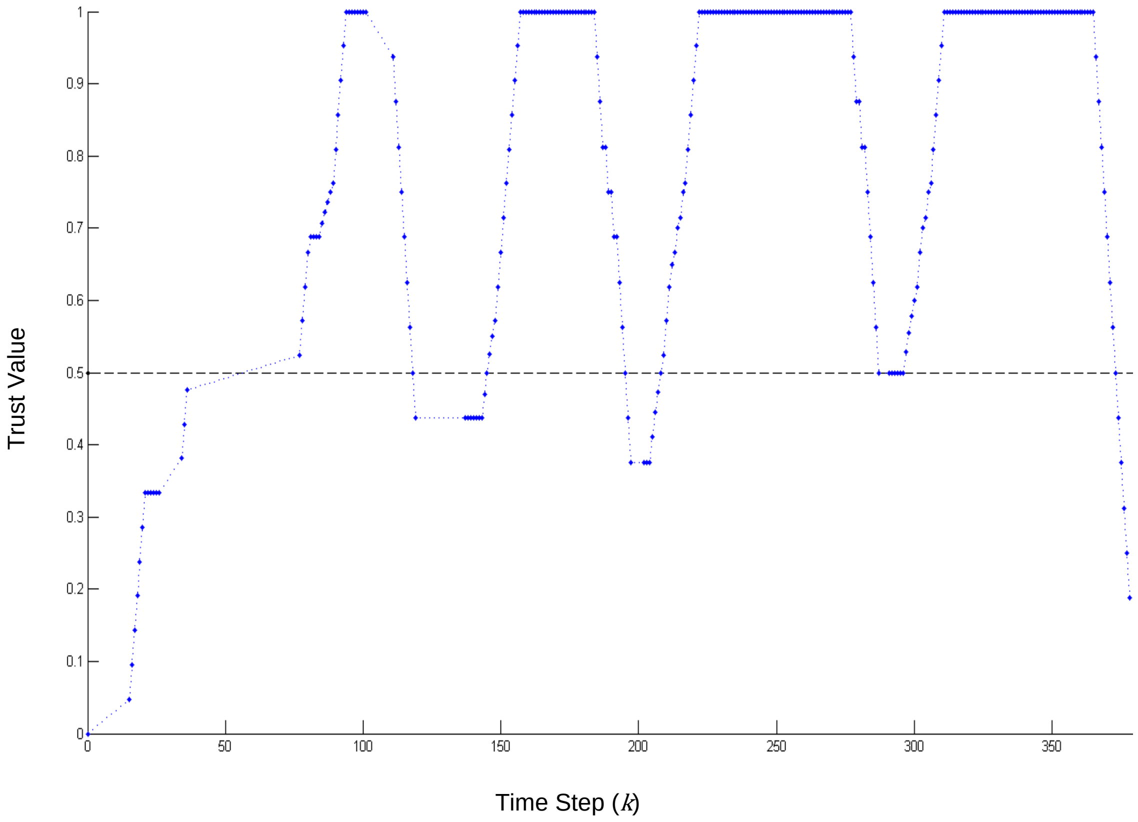
(d)

Figure 7.10 – Continued

to $W_{(3)}$ in order to ensure it continues to follow. The leader proceeds with this adjustment for 3 seconds until both the leader and the follower share the same current waypoint of $W_{(4)}$. At this point, the leader switches back to its default control mode, and attempts to stop at $W_{(3)}$. Unfortunately, because of the leader's momentum, the leader overshoots the position of $W_{(3)}$ at $k = 177$, causing the follower to make an unacceptable observation about its progress to $W_{(4)}$. Successive unfavorable observations by the follower eventually cause the follower to switch to its own default control at $k = 180$. Over the next 19 seconds, the follower attempts to correct its trajectory toward $W_{(4)}$, until it switches back to leader following mode at $k = 199$ due to 6 successive favorable observations of the leader. For the remainder of the path, the follower never switches back to its default control mode, preserving the togetherness of both vehicles. Figure 7.10d also shows an improvement in precision in switching waypoints, which occurs consistently at approximately 120 meters from the target waypoint.

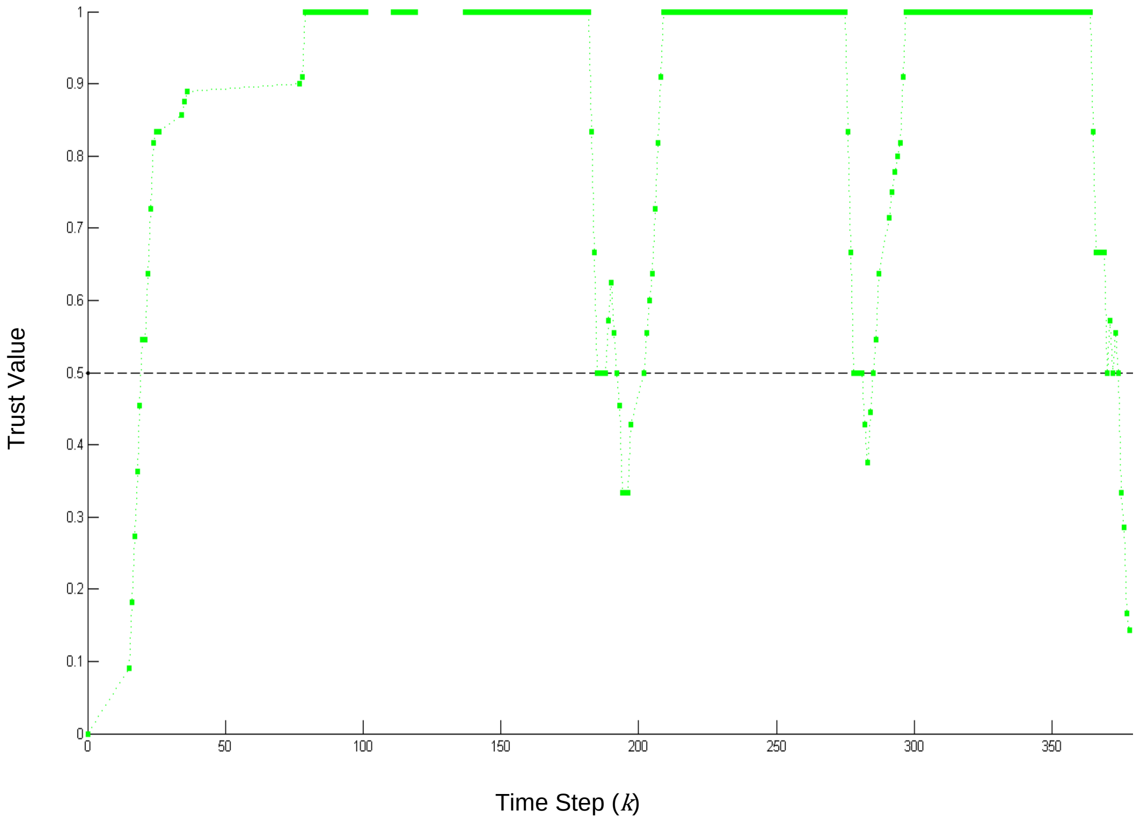
We see even more interesting simulation results in Figure 7.11 when the leader's Follower Acceptance Function is set to "CF Likes My Heading." In this case, the leader only adjusts its trajectory if the trusted follower's heading is not similar to the leader's heading. In order words, the leader is assuming that a misaligned follower may not be happy with its leadership, even if they both agree on the target waypoint. This, of course, may not reflect the reality of the follower's perceptions of the leader, and as such, represents a mismatch in leader/follower paradigms.

In Figure 7.11a, we see that the leader records its first unfavorable observation



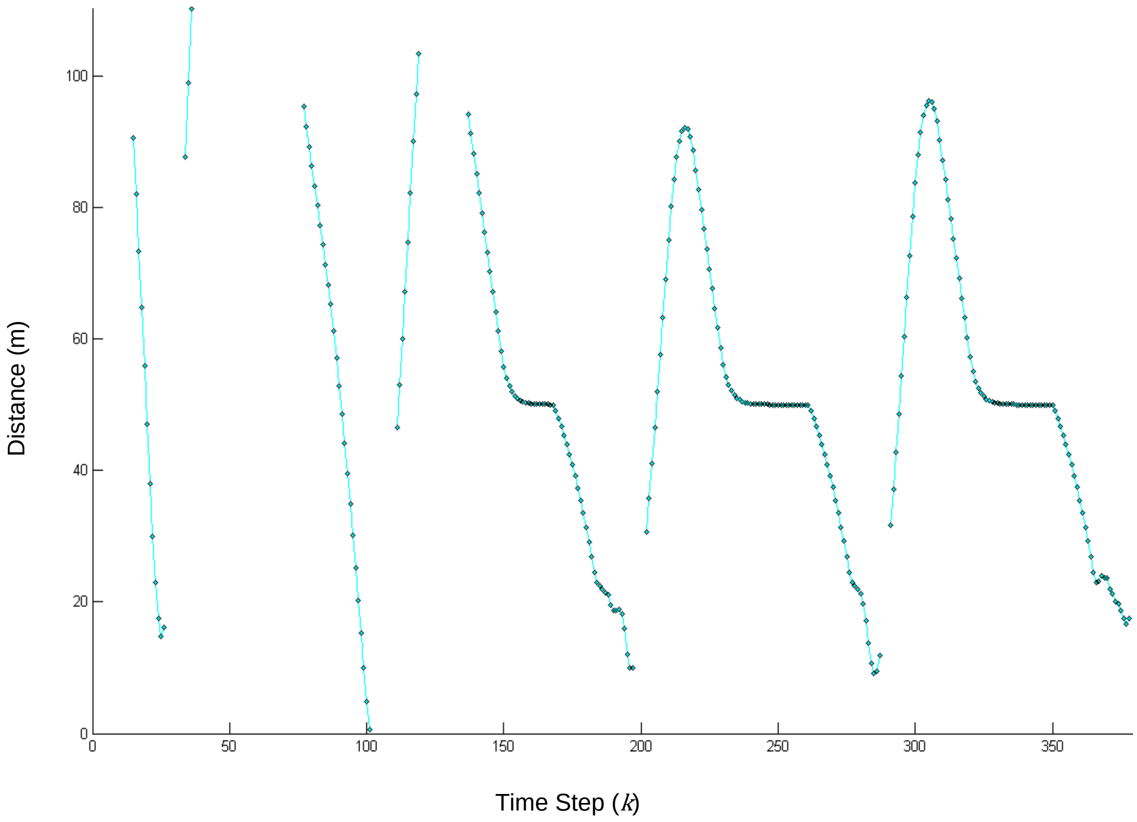
(a)

Figure 7.11. Simulation Results of Case Study 2 when Vehicle 1's Follower Acceptance Function is "CF Likes My Heading." The plot in (a) shows vehicle 1's trust towards vehicle 2 with respect to time. The plot in (b) shows vehicle 2's trust towards vehicle 1 with respect to time. The plot in (c) shows the distance between both vehicles with respect to time. The plot in (d) shows the distance between vehicle 2 and its current waypoint with respect to time. (This figure is presented in color; the black and white reproduction may not be an accurate representation.)



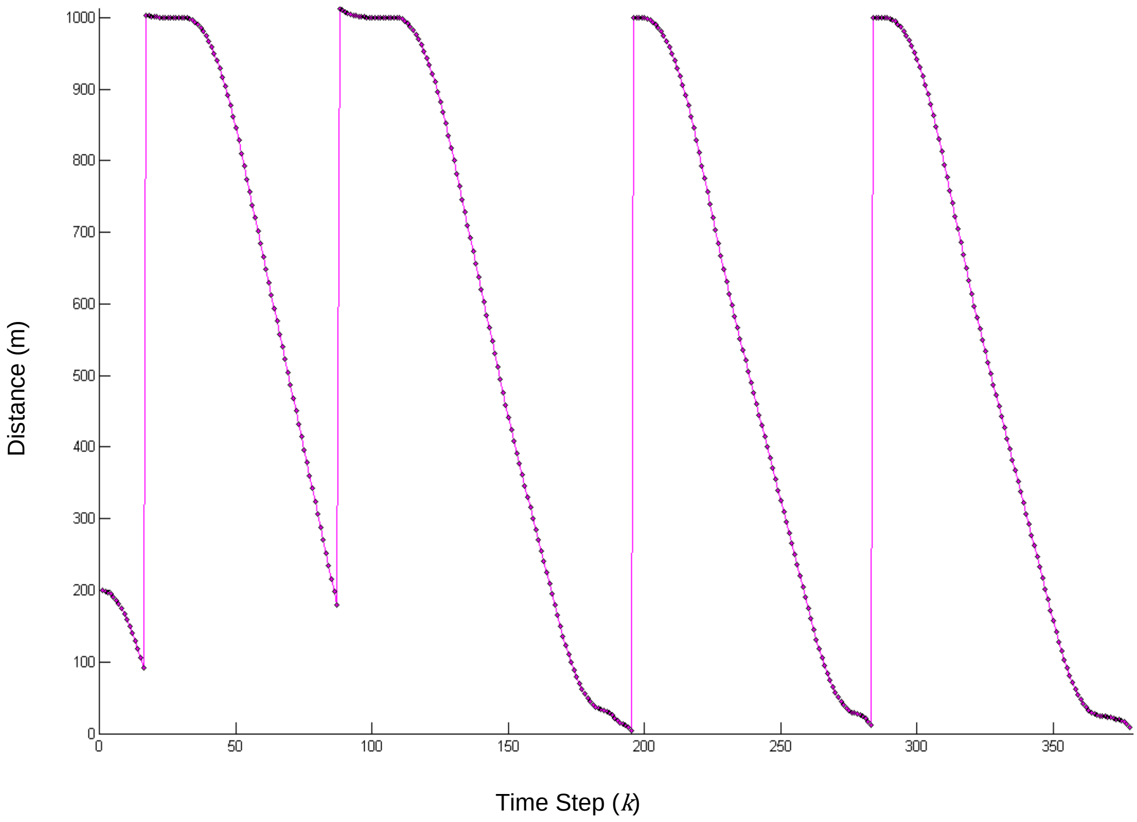
(b)

Figure 7.11 – Continued



(c)

Figure 7.11 – Continued



(d)

Figure 7.11 – Continued

of the follower at $k = 111$ because the follower's heading is misaligned with the leader's heading. That said, the follower at this point has complete trust in the leader and actually records a favorable observation (Figure 7.11b). Furthermore, both the leader and the follower share the same waypoint, $W_{(3)}$. As such, we can see that the leader's context of its follower is what is actually misaligned. The leader, in fact, fails to consider that the follower is more sluggish than the leader, with a maximum impulse force of 3600 N/s as opposed to the leader's 4000 N/s. And because of this, the follower is not able to turn as quickly as the leader, resulting in an apparent observation that the follower may not want to follow the leader.

This issue becomes more pronounced at the transition point between $W_{(3)}$ and $W_{(4)}$. At $k = 169$, the leader begins to brake to stop at $W_{(3)}$ and reaches a complete stop at $k = 184$. At $k = 185$, the leader turns toward $W_{(4)}$, but observes that it is misaligned with the follower. Given that the follower's current waypoint is still $W_{(3)}$, the leader attempts to turn around toward $W_{(3)}$. This results in somewhat of a jittery interaction between the two vehicles for several seconds, until the follower decides to break away from the leader at $k = 193$ and the leader decides to ignore the follower at $k = 196$. However, by $k = 203$, the follower is following and aligned with the leader again.

The results for this case show that the follower travelled a total of 4212.49 meters in 379 seconds within one revolution of the path, giving it an average speed of 11.11 meters per second – worse than the baseline case. But despite this, we see that the leader and the follower tended to stay close to each other, as exhibited by the continuity in Figures 7.11a, 7.11b, and 7.11c. We also see evidence of more precision and accuracy with respect to stopping at a waypoint in Figure 7.11d – less than 12 meters from the target waypoint.

In the end, however, we saw the best performance when the leader adjusted its trajectory in response to a target waypoint mismatch. This not only validates the feasibility of the trust-based controller, but it also reinforces the importance of vehicles sharing similar contexts in cooperative control strategies.

7.3.4 Case Study 3: Bad Vehicle

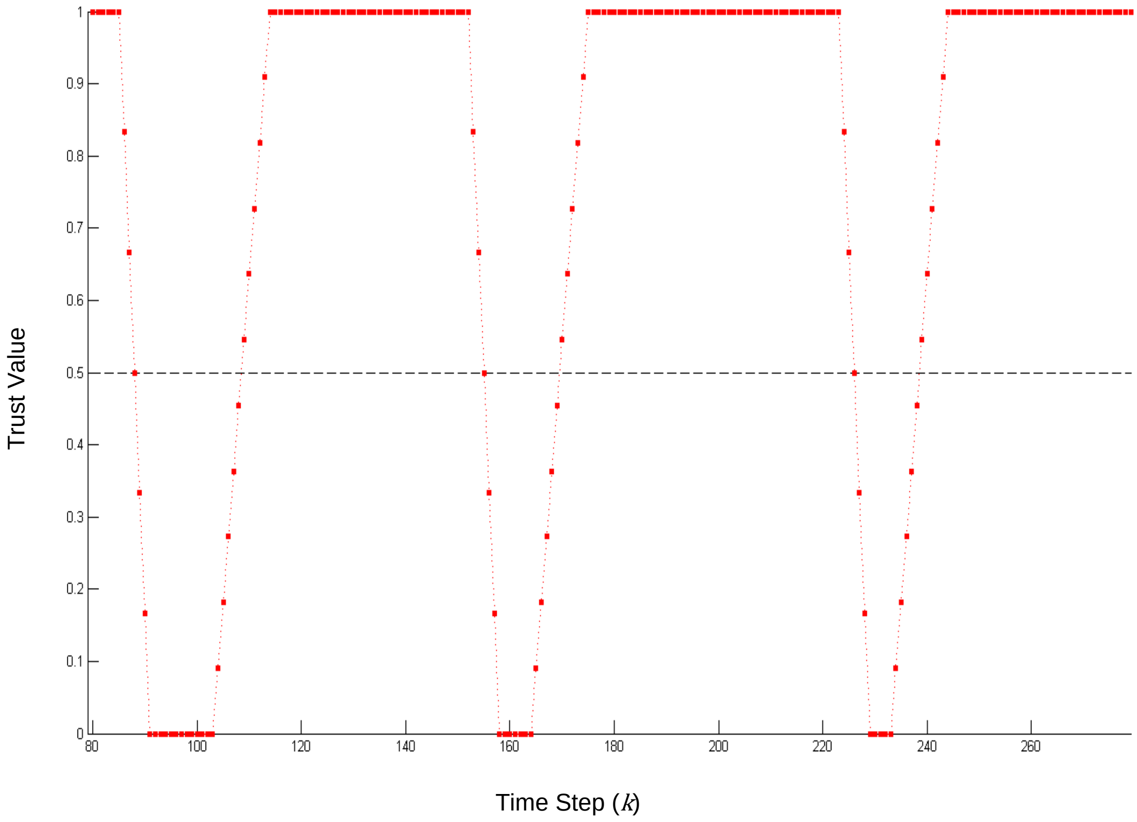
The purpose of our final case study is to demonstrate that trust-based controller can correct adjust to the behavior of a bad vehicle in the convoy. Bad behavior in this study refers to actively misleading followers. We set up the scenario with four vehicles, configured according to Table 7.8, using the default square waypoint configuration in Equation 7.46. The variable in this case study is the selection of the bad vehicle in the convoy. Our study will vary the selection of the bad vehicle and observe the behavioral changes in the normal vehicles.

To begin, we present our simulation results for the baseline case in Figure 7.12, where there are no bad vehicles in the convoy. Figure 7.12a and 7.12b show the trust value plots for vehicle 2 and 4, respectively, for their closest leaders. Given that the context for both vehicles considers the heading of their local leader with respect to the ideal heading towards the current waypoint, we notice a characteristic loss of trust at the transitions between waypoints. However, this trust is regained along the straight-aways between waypoints, ensuring stability in the convoy. Figure 7.12c shows the path trace

Table 7.8.

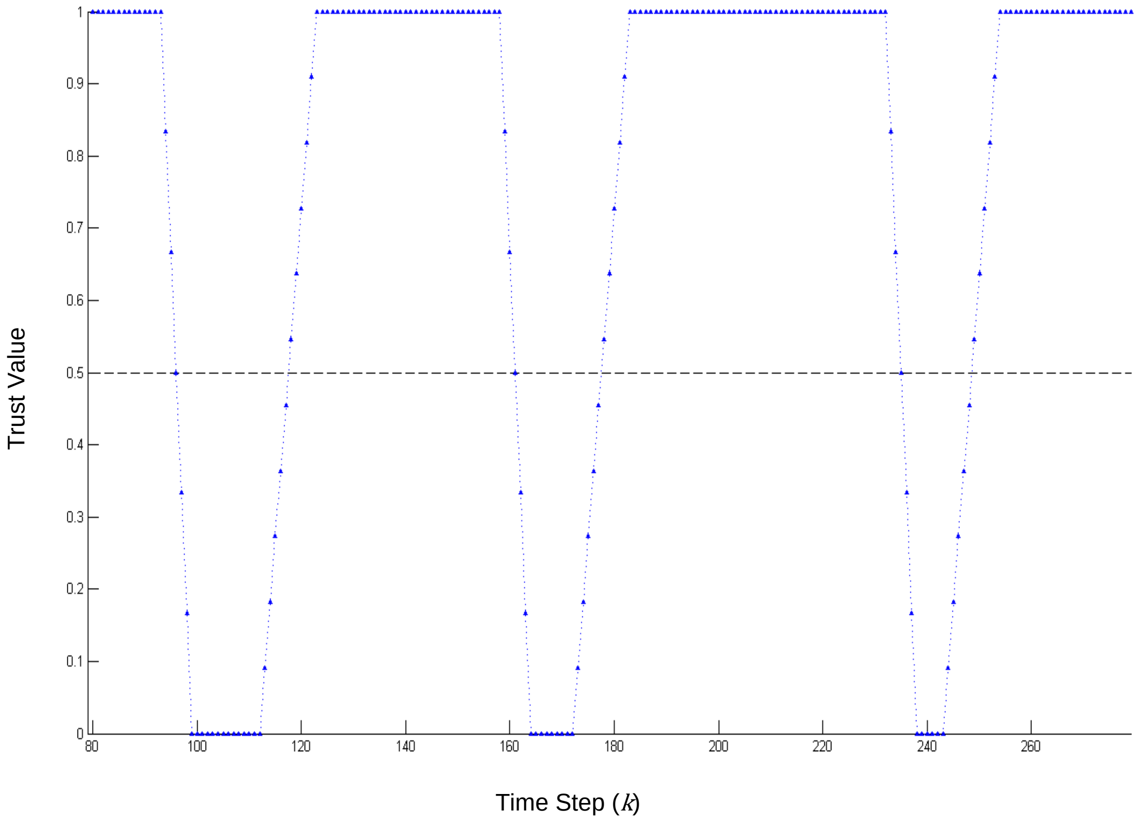
Case Study 3 Vehicle Configurations

Parameter	**Vehicle 1**	**Vehicle 2**	**Vehicle 3**	**Vehicle 4**
Mass	4000 kg	4000 kg	4000 kg	4000 kg
Min Following Distance	50 m	50 m	50 m	50 m
Max Speed	15 m/s	18 m/s	18 m/s	18 m/s
Max Impulse Force	3500 N/s	3500 N/s	3500 N/s	3500 N/s
Sensor Range	100 m	100 m	100 m	100 m
Sensor HFOV	110°	110°	110°	110°
Control Scheme	**{Always Moving, Bad Vehicle}**	Always Moving	**{Always Moving, Bad Vehicle}**	Always Moving
Color	Red	Green	Blue	Cyan
Leader Acceptance Function	CL Always Wrong	CL Heading To My WP	CL Heading To My WP	CL Heading To My WP
Leader Trust Threshold	0.5	0.5	0.5	0.5
Leader Tolerance	5	5	5	5
Leader Confirmation	10	10	10	10
Follower Acceptance Function	CF Dislikes Everything	CF Dislikes Everything	CF Dislikes Everything	CF Dislikes Everything
Follower Trust Threshold	0.5	0.5	0.5	0.5
Follower Tolerance	5	5	5	5
Follower Confirmation	10	10	10	10



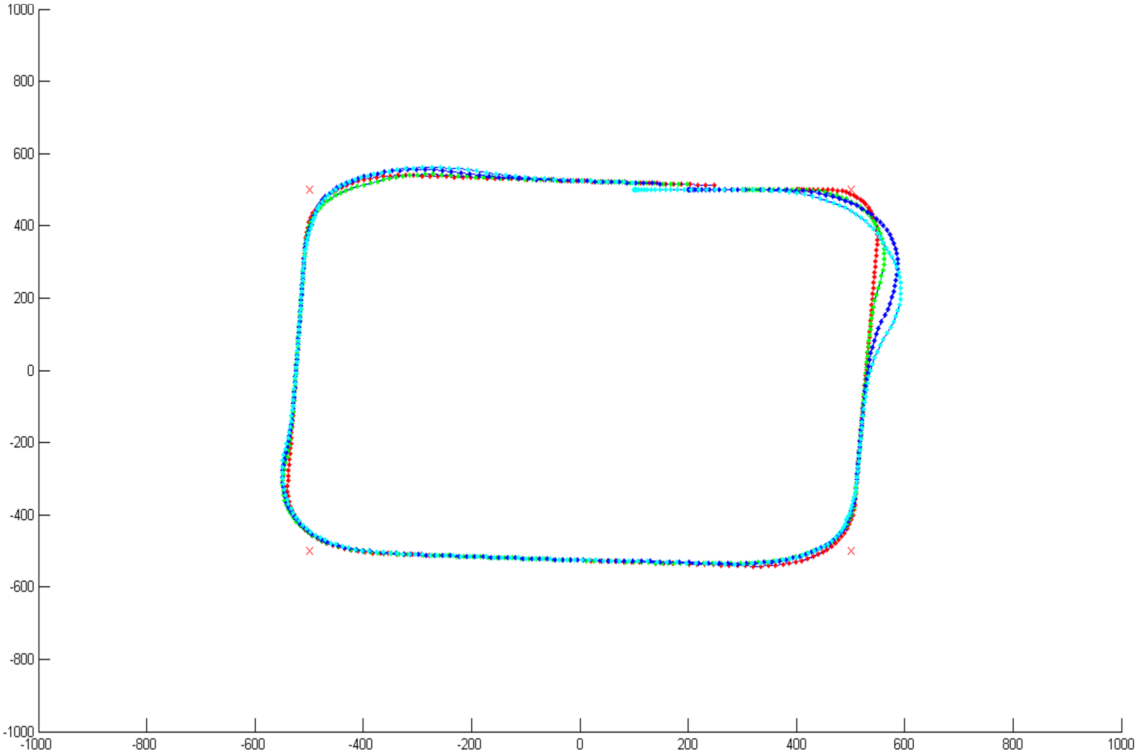
(a)

Figure 7.12. Simulation Results of Case Study 3 with No Bad Vehicles. The plot in (a) shows the trust value for vehicle 2's closest leader with respect to time. The plot in (b) shows the trust value for vehicle 4's closest leader with respect to time. The plot in (c) shows the path trace of each vehicle. (This figure is presented in color; the black and white reproduction may not be an accurate representation.)



(b)

Figure 7.12 – Continued



(c)

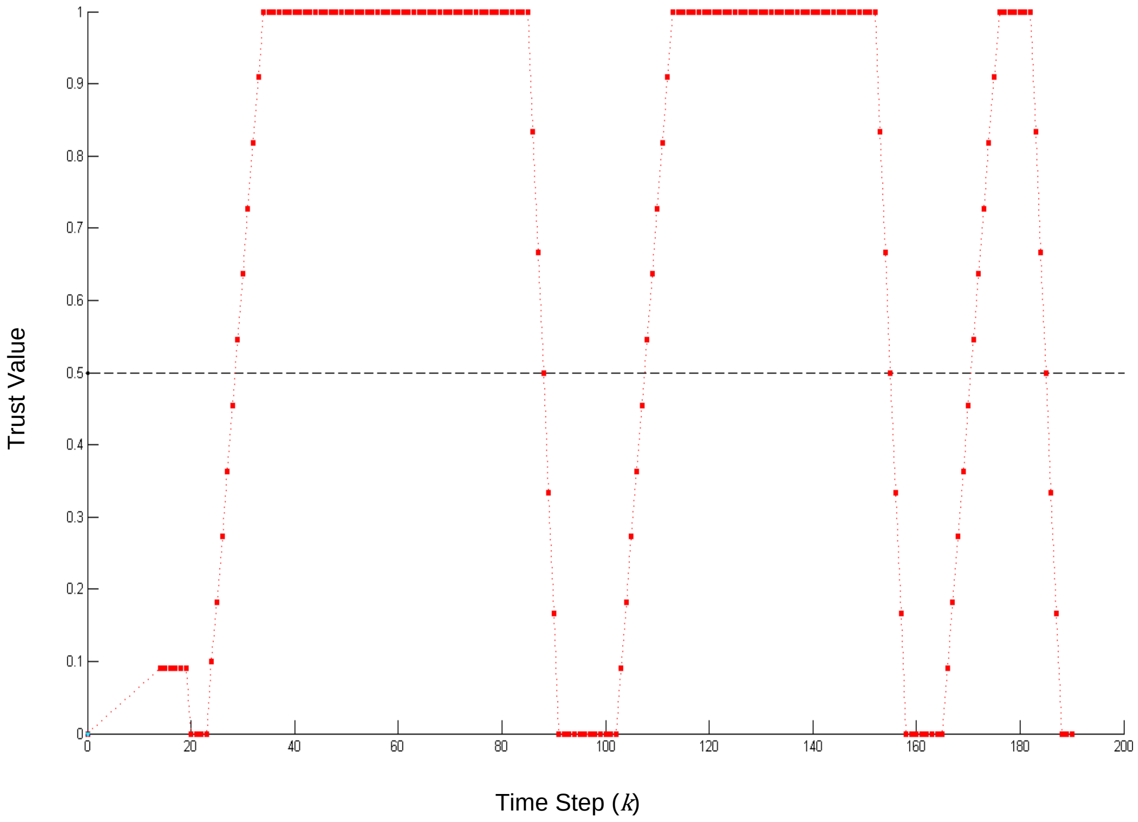
Figure 7.12 – Continued

of each vehicle for one revolution of the path.

Next, we set vehicle 1 – the leader of the convoy – to be the only bad vehicle. Our results for this case are displayed in Figure 7.13. The data logs show that vehicle 1 starts its bad behavior at $k = 175$. However, vehicle 2 does not detect the bad behavior until $k = 183$. However, due to its low tolerance for unfavorable observations, vehicle 2 quickly switches to its default control mode at $k = 186$. It is important to note that the data logs show that neither vehicle 3 nor vehicle 4 ever detected any problem with vehicle 1 or vehicle 2.

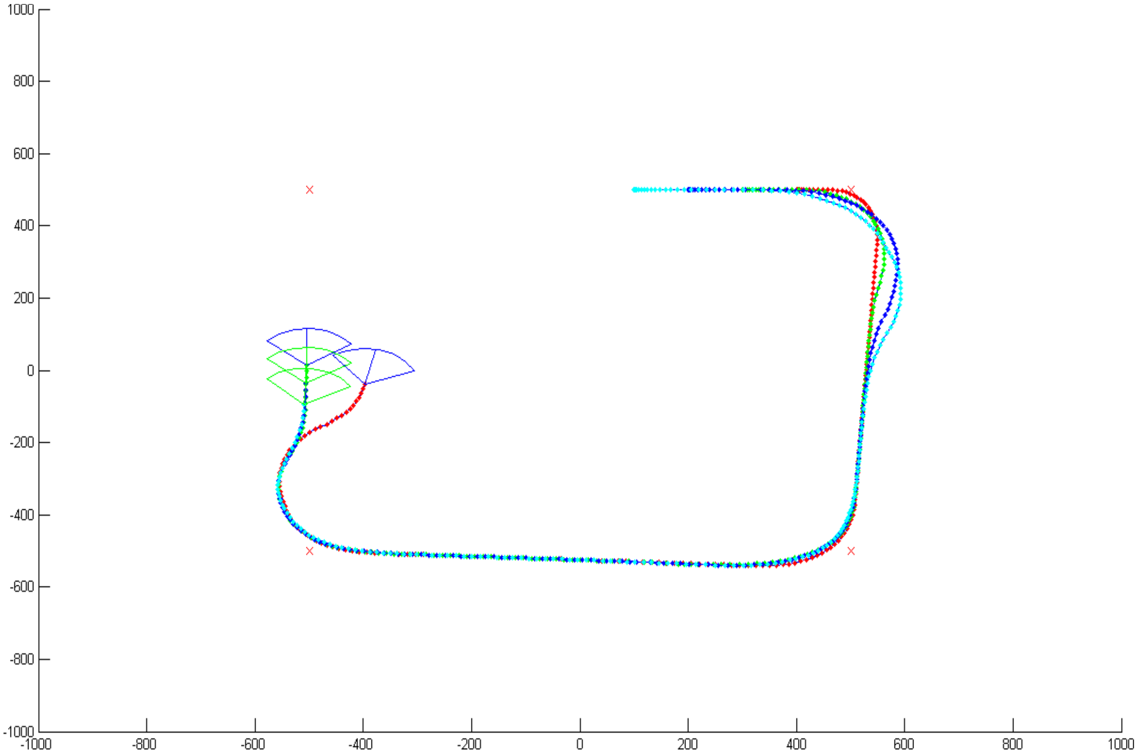
Finally, we set vehicle 3 to be the only bad vehicle in the convoy. Our results for this case are displayed in Figure 7.14. The data logs show that vehicle 3 begins behaving badly at $k = 59$. However, vehicle 4 does not detect the bad behavior until $k = 67$. But, due to its low tolerance for unfavorable observations, vehicle 4 quickly switches to its default control mode at $k = 70$. It remains in this mode until $k = 121$ when it switches to leader following mode by following vehicle 2.

In both cases, the normal vehicle detects and avoids the bad vehicle using its trust-based controller. However, more work to improve the trust-based controller can be done for more realistic scenarios. Other bad behaviors, such as stopping unexpectedly, moving slower than the convoy tempo, or unexpectedly lengthening the minimum following distance can also disrupt military convoy operations. That said, apparent bad behaviors may potentially be correct behaviors under certain circumstances, such as if a vehicle detects a roadside bomb or if a convoy is driving through pedestrian-filled roads. Thus, the importance of framing observations in the proper context cannot be understated when designing an actual trust-based controller.



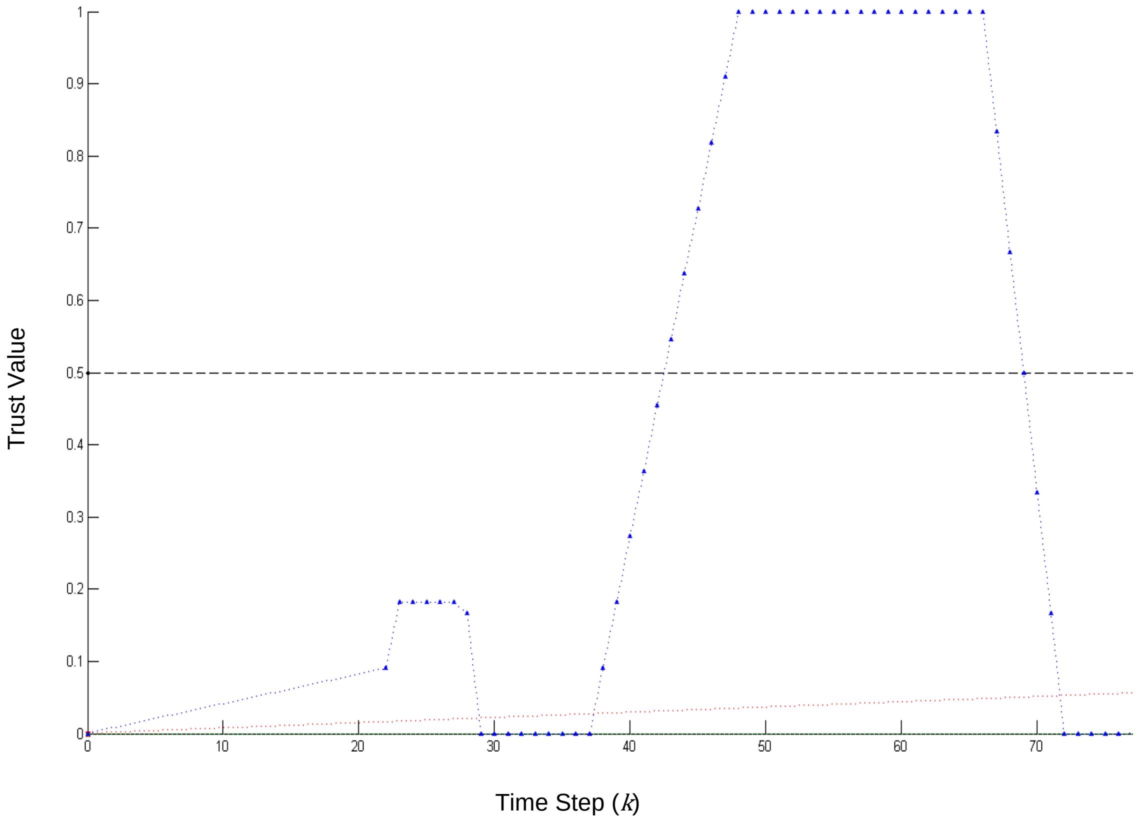
(a)

Figure 7.13. Simulation Results of Case Study 3 with Vehicle 1 as the Bad Vehicle. The plot in (a) shows the trust value for vehicle 2's closest leader with respect to time. The plot in (b) shows the path trace of each vehicle. (This figure is presented in color; the black and white reproduction may not be an accurate representation.)



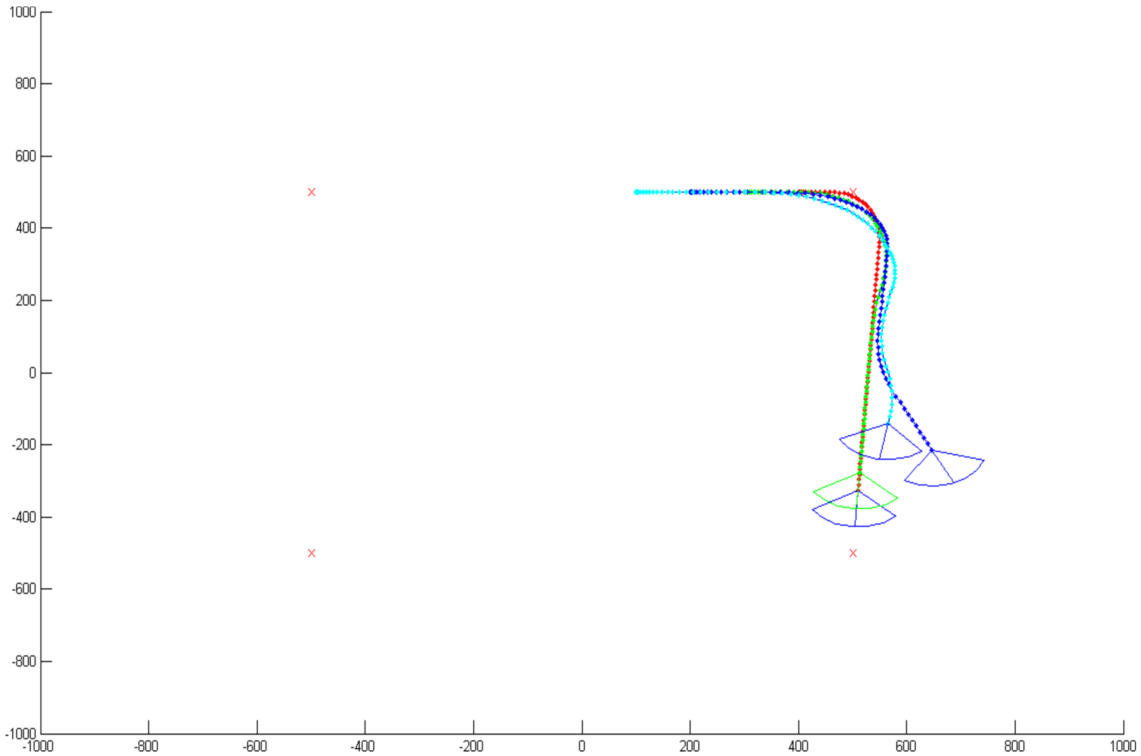
(b)

Figure 7.13 – Continued



(a)

Figure 7.14. Simulation Results of Case Study 3 with Vehicle 3 as the Bad Vehicle. The plot in (a) shows the trust value for vehicle 4's closest leader with respect to time. The plot in (b) shows the path trace of each vehicle. (This figure is presented in color; the black and white reproduction may not be an accurate representation.)



(b)

Figure 7.14 – Continued

Fortunately, developers can leverage the primary benefit of the trust-based controller design in this dissertation – its deliberate separation of the context (acceptance functions and RoboTrust) and control algorithms (leader following, leading follower, and default control modes). This fact is pervasively demonstrated in all of the case studies. Indeed, neither the context nor the control algorithms need to be aware of each other, giving developers the ability to independently verify and validate correct contexts and control prior to integrating both for intelligent behaviors.

Conclusion

In this chapter, we studied decentralized autonomous convoy operations. We performed a theoretical analysis of a decentralized convoy using cooperative trust game theory and discovered that the trust payoff can be maximized if agents treat immediate leaders and followers as surrogates for the system of agents in front and behind them, respectively. Thus, rather than treating the decentralized convoy as a single trust game (as we did in Chapter Three), we consider it to be a collection of many 2-agent convoy trust games, where the players in each game consist of a super-agent leader and a super-agent follower. Then, we developed a simulated trust-based vehicle controller for decentralized autonomous convoy operations. The controller switches automatically between three modes of operation (leader-following, leading-follower, and default control) based on its cultivated trust in its immediate leaders and followers for its selected contexts. In our case studies, we demonstrated mode switching between leader-following and default control, as well as leading-follower and default control. In addition, we demonstrated the trust-based controller correctly responding to badly behaving vehicles during a simulated convoy mission. As such, we can conclude that the trust-based controller using the RoboTrust model is a feasible candidate solution for soft security during a convoy mission. The design of the trust-based controller, in particular, lends itself well for implementation on robotic platforms, given its modularization between the context (acceptance functions and RoboTrust) and control algorithms.

CHAPTER EIGHT

CONCLUSIONS AND FUTURE WORK

8.1 Conclusions

When an interaction with another involves uncertainty, a person will leverage trust to simplify expectations and justify decisions that trade-off security and performance. It is especially true in war. Interpersonal interactions with high uncertainty are prevalent during wartime, and warfighters have learned to rely on trust within their teams to not only overcome personal fear, but also to maintain unit cohesion. This helps them to maximize both individual and group effectiveness in meeting mission goals and objectives.

In this dissertation, we postulated that analogous trust concepts could also apply to military robotic systems working in cooperative teams. Often, we assume that robots within the same team should be regarded as being fully trustworthy for cooperative tasks because of the complexity involved in producing robust, autonomous multi-robot solutions. However, military robots have unique vulnerabilities related to their exposure to cyber attacks, which may not only increase the risk to mission success, but also endanger the lives of friendly forces. Hard security mechanisms, such as cryptography protection and authentication protocols, are vital to minimizing this exposure. However, hard security mechanisms cannot protect against illegitimate behaviors after a hard security event, such as decryption or identification validation. Trust models, which dynamically adjust to observed behaviors or recommendations, can mitigate the risks of illegitimate behaviors, and therefore, can be used to defend against this type of threat.

Our research goal in this dissertation was to determine the feasibility of computational trust as a defensive capability against unacceptable behaviors in military autonomous systems. To meet this goal, we sought to develop new or improved algorithms and frameworks for trust cultivation, aggregation, and propagation in distributed networked teams. But to narrow our scope, we directed our application focus toward autonomous military convoy operations.

Our first significant technical result was the development of the cooperative trust game – a new and formal mathematical framework to predict coalition formation and disbanding in response to trust-based interactions. This framework was developed on top of cooperative game theory, which allowed us to show how each agent's trust preferences in a game can influence a group's ability to reason about trustworthiness. The trust payoff value in cooperative trust game is calculated as the difference between trust synergy and trust liability.

As part of our research, we characterized different classes of cooperative trust games, provided a general model for cooperative trust games, and applied the model to autonomous convoy operations within the context of moving forward together. With respect to this application, we proved that the most optimal trust payoff occurs when the lead vehicle acts as the trusted third-party between all of the follower vehicles. This result, however, assumes that all vehicles within a convoy have the ability to interact with each other at all times, which may not be true in practice. As such, we also applied the cooperative trust game to a decentralized convoy network, and discovered that the trust payoff can be maximized if agents view immediate leaders and followers as surrogates for the whole system of agents in front and behind them, respectively. The maximum potential trust payoff for decentralized convoys, however, is capped at 1, regardless of the number of vehicles in the convoy. Centralized convoys, on the other hand, can reach a maximum potential trust payoff equal to $|N| - 1$.

Our second significant technical result was the development of the RoboTrust model, which calculates trustworthiness as the smallest value in a set of maximum-likelihood estimates that are based on different historical observations. One of its key features is its explicit separation between the context of an observation and the actual trust calculation, which allows developers to focus engineering efforts to correctly describe contexts without needing to understand how users intend to interpret the trust calculation from an observation history. In our work, contexts are described by acceptance functions, which decide whether or not an agent should collectively deem a set of observed data as acceptable. The series of results from an acceptance function describes an acceptance observation history, which is then used within the RoboTrust model to evaluate the level of trust. RoboTrust is natively designed to use directly acquired observation histories as an input to calculate trust; however, we also provided an extension which can use indirectly-acquired observation histories (recommendations) from multiple first-neighbors to evaluate trustworthiness in unobservable agents

After conducting a series of performance tests with RoboTrust and two other commonly-used trust models, we integrated the RoboTrust model within solutions for two different problems, namely the consensus problem and the autonomous convoy soft security problem. The consensus problem was intended to represent a traditional multi-agent "controls" application while the autonomous convoy soft security problem was intended to represent a traditional multi-agent "decision" application.

For the consensus problem, we provided a distributed, discrete-time, trust-based consensus protocol and proved its asymptotic convergence to an agreement space. We then analyzed the protocol under different experiments of static-trust and dynamic-trust in a simple three-agent network, and discovered some interesting findings. Our static-trust experiment indicated an inverse correlation between the overall level of trust in a network and convergence time. Our dynamic-trust experiments, which used the RoboTrust model to calculate trust, indicated that higher tolerance values tended to shorten convergence time, higher confirmation values tended to extend convergence time, and a higher tolerance and confirmation parameter diversity produced more variety in final consensus results for both the final value and convergence time.

For the autonomous convoy security problem, we developed a simulated trust-based vehicle controller for decentralized convoy operations and demonstrated its operations within a series of case studies. We showed control mode switching based on trust levels between leader-following and default control as well as leading-follower and default control. In addition, we demonstrated how the controller can correctly respond to badly behaving vehicles during a simulated convoy mission. From our results, we conclude that the trust-based controller using the RoboTrust model is a feasible candidate solution for soft security during a convoy mission. In particular, the controller's modularization between the context and control algorithms lends itself well for implementation on existing robotic platforms.

8.2 Future Work

Having shown the feasibility of the RoboTrust model within a simulated autonomous convoy controller, future work will be directed toward integrating trust-based control within an actual military robotic demonstration platform. Planned efforts within the next year in support of this goal include the following:

1. Within a FY2014 Technology Program Agreement (TPA) with U.S. Army Tank-Automotive Research Development and Engineering Center (TARDEC) and U.S Army Research Laboratory (ARL), TARDEC and ARL researchers will validate various trust models under different relevant contexts to improve security in heterogeneous multi-agent systems.
2. With a university seed money award from the U.S. Army National Automotive Center (NAC) to University of Texas in Arlington (UTA), UTA investigators will develop a working proof-of-concept trust-based controller for a small autonomous robot. The project will use RoboTrust as the basis for cultivating trust and implement it within the Robotic Operating System (ROS). In addition, UTA plans leverage funding from the Air Force Office of Scientific Research (AFOSR) European Office of Aerospace Research and Development (EOARD) to incorporate unmanned aerial vehicles within this project for cooperative teaming experiments.

There are also plans for more general short-term efforts to support the basic research endeavors connected to the RoboTrust model.

1. With the expected funding for FY2014 In-House Laboratory Innovative Research (ILIR) awards from the Office of the Assistant Secretary of the

Army for Acquisition, Logistics and Technology (ASA(ALT)), TARDEC plans to pursue to two basic research projects. The first project will use the RoboTrust model to perform a theoretical analysis of the impact of topology and dynamics on trust aggregation and propagation in order to obtain insights on the fundamental limits of trust estimation accuracy. The second project will incorporate the RoboTrust model into the basic particle swarm optimization (PSO) algorithm, thereby being the first research effort to connect the computational trust research domain with the multi-dimensional optimization research domain.

2. As an internal TARDEC research exercise, a team of TARDEC researchers will use RoboTrust within various diagnostics algorithms to help detect abrupt persistent faults, drift (incipient) faults, and abrupt intermittent faults. The algorithms will be tailored for the Electrical Power System (EPS) testbed at the National Aeronautics and Space Administration (NASA) Ames Research Center Advanced Diagnostics and Prognostics Testbed (ADAPT) laboratory. The EPS is a hardware test-bed that is used for studying the electrical power and distribution systems for satellites. It illustrates many of the design properties in real-world satellites, such as controllability, observability, and redundancy.

Unplanned, but potential basic research endeavors also exist for our work. These are listed to inspire the imagination of researchers who wish to expand on the work in this dissertation.

1. **Extend the Cooperative Trust Game**. Develop new or improved definitions for trust synergy and trust liability with respect to the cooperative trust game.
2. **Apply the Cooperative Trust Game**. Develop new **Σ** and **Λ** matrices for different problems and analyze the results using the techniques in this dissertation.
3. **Extend the RoboTrust Class of Models**. Develop new RoboTrust models based on different probability distributions, such as Gamma or Poisson distributions.
4. **Learn the Acceptance Function**. Describe acceptance functions in other ways, such as neural networks or support vectors, and learn (either supervised or unsupervised) the acceptance regions for a particular context.
5. **Learn $(\boldsymbol{\tau}, \boldsymbol{c})$ Pairs Online**. Develop ways to learn or tune (τ, c) pairs for the RoboTrust model automatically online, in response to changes in other agents or the environment.
6. **Integrate RoboTrust into Diverse Research Domains**: Experiment using trust to gauge behavioral uncertainty in different research domains, such as optimization, diagnostics/prognostics, sensor fusion, machine learning, pattern recognition, or control.

REFERENCES

[1] A. Abdul-Rahman and S. Hailes, "Supporting Trust in Virtual Communities," in *33rd Hawaii International Conference on Systems Science*, 2000.

[2] S. Ackerman. (2012, September) The Pentagon Doesn't Trust Its Own Robots. [Online]. http://www.wired.com/dangerroom/2012/09/robot-autonomy/

[3] W. J. Adams and N. J. Davis, "Toward a Decentralized Trust-Based Access Control System for Dynamic Collaboration," in *Proc. 6th IEEE SMC Information Assurance Workshop*, West Point, 2005, pp. 317-324.

[4] B. Adams and R. Webb, "Trust in Small Military Teams," in *7th International Command and Control Technology Symposium*, 2002.

[5] E. Ahmed, K. Samad, and W. Mahmood, "Cluster-Based Intrusion Detection (CBID) Architecture for Mobile Ad Hoc Networks," in *AusCERT Asia Pacific Information Technology Security Conference*, Gold Coast, 2006.

[6] P. Albers et al., "Security in Ad Hoc Networks: A General Intrusion Detection Architecture Enhancing Trust Based Approaches," in *Proc. 1st International Workshop on Wireless Information Systems*, 2002, pp. 1-12.

[7] C. E. Alchourrón, P. Gärdenfors, and D. Makinson, "On The Logic of Theory Changes: Partial Meet Contraction and Revision Functions," *Journal of Symbolic Logic*, vol. 50, pp. 510-530, 1985.

[8] D. C. Arney and E. Peterson, "Cooperation in social networks: communication, trust, and selflessness.," in *26th Army Science Conference*, Orlando, FL, 2008.

[9] R. Axelrod, *The Evolution of Cooperation: Revised Edition*. New York, NY: Basic Books, 2006.

[10] Y. Bachrach, A. Parnes, D. Procaccia, and J. Rosenschein, "Gossip-based aggregation of trust in decentralized reputation systems," *Autonomous Agents and Multi-Agent Systems*, vol. 19, no. 2, pp. 153-172, 2009.

[11] A. C. Baier, "Trust and Antitrust," *Ethics*, vol. 96, pp. 231-260, 1986.

[12] J. Baker, "Trust and Rationality," *Pacific Philosophical Quarterly*, vol. 68, pp. 1-13, 1987.

[13] S. Balfe, P. Yau, and K. G. Paterson, "A Guide to Trust in Mobile Ad Hoc Networks," *Security and Communication Networks*, vol. 3, no. 6, pp. 503–516, 2010.

[14] P. Ballal and F. Lewis, "Trust-Based Collaborative Control for Teams in Communication Networks," in *26th Army Science Conference*, Orlando, FL, 2008.

[15] A. Baltag and S. Smets, "Dynamic Belief Revision over Multi-Agent Pausibility Models," in *Proc. 7th Conference on Logic and the Foundations of Game and Decision Theory*, Liverpool, 2006.

[16] J. Baras, T. Jiang, and P. Purkayastha, "Constrained Coalitional Games and Networks of Autonomous Agents," in *ISCCSP 2008*, Malta, 2008, pp. 972-979.

[17] L. C. Becker, "Trust as Noncognitive Security about Motives," *Ethics*, vol. 107, no. 1, pp. 43-61, 1996.

[18] J. Bennett. (2013, April) The Mutter Backdoor: Operation Beebus with New Targets. [Online]. http://www.fireeye.com/blog/technical/malware-research/2013/04/the-mutter-backdoor-operation-beebus-with-new-targets.html

[19] R. Branzei, D. Dimitrov, and S. Tijs, "A new characterization of convex games," in *Tiburg University, Center of Economic Research*, 2004.

[20] G. Brewka, I. Niemela, and M. Truszczynski, "Nonmonotonic Reasoning," in *Handbook of Knowledge Representation*, F. van Harmelen, V. Lifschitz, and B. Porter, Eds. St. Louis, MO: Elsevier, 2007.

[21] S. Brodt and M. A. Korsgaard, "Group Identity and Attachment: Two Paths to Trust and Cooperation in Groups," in *16th IACM Conference*, Melbourne, 2003.

[22] C. Burnett, T. Norman, and K. Sycara, "Trust Decision-Making in Multi-Agent Systems," in *22nd International Joint Conference on Artificial Intelligence*, Barcelona, 2011, pp. 115-121.

[23] F. Cajori, *Sir Isaac Newton's Principia, Vol. II The System of the World*. Berkeley and Los Angeles, CA: University of California Press, 1934.

[24] L. Capra and M. Musolesi, "Autonomic Trust Prediction for Pervasive Systems," in *Proc. 20th International Conference on Advanced Information Networking and Applications*, 2006, pp. 481-488.

[25] C. Castelfranchi and R. Falcone, "Principles of trust for MAS: cognitive anatomy, social importance, and quantification," in *International Conference on Multi-Agent Systems (ICMAS98)*, 1998, pp. 72-79.

[26] C. Castelfranchi and R. Falcone, "Social trust: a cognitive approach," *Trust and Deception in Virtual Societies*, pp. 55-90, 2000.

[27] C. Castelfranchi and R. Falcone, "Trust is much more than subjective probability: mental components and sources of trust," in *33rd Hawaii International Conference on System Sciences (online edition)*, vol. 6, 2000.

[28] P. Chandler and M. Pachter, "Challenges," in *UAV Cooperative Decision and Control*. Philadelphia, PA: Society for Industrial and Applied Mathematics (SIAM), 2009, pp. 15-35.

[29] B. J. Chang and S. L. Kuo, "Markov Chain Trust Model for Trust Value Analysis and Key Management in Distributed Multicast MANETs," *IEEE Transactions on Vehicle Technology*, vol. 58, no. 4, pp. 1846-1863, May 2009.

[30] K. Chan, A. Swami, Q. Zhao, and A. Scaglione, "Consensus Algorithms over Fading Channels," in *Proc. MILCOM 2010*, San Jose, 2010, pp. 549-554.

[31] J. Cho, A. Swami, and I. Chen, "A Survey on Trust Management for Mobile Ad Hoc Networks," *IEEE Communications Surveys and Tutorials*, vol. 13, no. 4, pp. 562-583, 2011.

[32] S. Choudhury, S. D. Roy, and S. A. Singh, "Trust Management in Ad Hoc Network for Secure DSR Routing," *Novel Algorithms and Techniques in Telecommunications, Automation, and Industrial Electronics*, pp. 496-500, 2008.

[33] P. R. Cohen and H. J. Levesque, "Intension is choice with commitment," *Artificial Intelligence*, vol. 42, no. 2-3, pp. 213-261, 1990.

[34] V. Conitzer and T. Sandholm, "Computing Shapley values, manipulating value division schemes, and checking core membership in multi-issue domains," in *AAAI Conference on Artificial Intelligence*, 2004.

[35] S. Corson and J. Macker, "Mobile Ad Hoc Networking (MANET): Routing Protocol Performance Issues and Evaluation Considerations," in *RPC 2501 (Informational)*, 1999.

[36] J. M. Dasch and D. J. Gorsich, *The TARDEC Story, Sixty-five Years of Innovation, 1946-2010*, 1st ed., M. Roddin and R. van Enkenvoort, Eds.: U.S. Army Research, Development and Engineering Command, 2012.

[37] N. Daukas, "Epistemic Trust and Social Location," *Episteme*, vol. 3, no. 1-2, pp. 109-124, 2006.

[38] R. de Sousa, *The Rationality of Emotion*. Cambridge: MIT Press, 1987.

[39] "Department of Defense Strategy for Operating in Cyberspace," U.S. Department of Defense, 2011.

[40] A. Dixit and B. Nalebuff, "Prisoners' Dilemmas and How to Resolve Them," in *The Art of Strategy*. New York: W.W. Norton and Company, 2008, pp. 64-101.

[41] E. Dougherty and M. Bittner, "Causality, Randomness, Intelligibility, and the Epistemology of the Cell," *Current Genomics*, vol. 11, pp. 221-237, 2010.

[42] P. England, Q. Shi, R. J. Askwith, and F. Bouhafs, "A Survey of Trust Management in Mobile Ad-Hoc Networks," in *Proc. 13th PGNET*, Liverpool, 2012.

[43] R. Falcone, G. Pezzulo, and C. Castelfranchi, "A fuzzy approach to a belief-based trust computation," in *Lecture Notes on Artificial Intelligence*, 2003, pp. 73-86.

[44] P. Faulkner, "A Genealogy of Trust," *Episteme*, vol. 4, no. 3, pp. 305-321, 2007.

[45] T. Fowler, *Bacon's Novum Organum*. London, UK: MacMillan and Co., 1878.

[46] K. Fullam and K. Barber, "Dynamically learning sources of trust information: experience vs. reputation," in *6th International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, Honolulu, 2007, pp. 1062–1069.

[47] G. Galilei, *Dialogues Concerning Two New Sciences*, 2nd ed., S. Drake, Ed. Toronto, ON: Wall & Emerson, Inc., 1989.

[48] D. Gambetta, "Can We Trust Trust?," in *Trust: Making and Breaking Cooperative Relations*, D. Gambetta, Ed. Oxford: Basil Blackwell, 1990, pp. 213-237.

[49] D. Gambetta, Ed., *Trust: Making and Breaking Cooperative Relations*. Oxford, UK: Basil Blackwell, 1990.

[50] P. Gärdenfors and H. Rott, "Belief Revision," in *Handbook of Logic in Artificial Intelligence and Logic Programming*. Oxford, UK: Oxford University Press, 1995, pp. 35-132.

[51] T. Ghosh, N. Pissinou, and K. Makki, "Towards Designing a Trust Routing Solution in Mobile Ad Hoc Networks," *Mobile Networks and Applications*, vol. 10, pp. 985-995, 2005.

[52] S. Goering, "Postnatal Reproductive Autonomy: Promoting Relational Autonomy and Self-Trust in New Parents," *Bioethics*, vol. 23, no. 1, pp. 9-19, 2009.

[53] A. I. Goldman, "Internalism Exposed," *Journal of Philosophy*, vol. 96, no. 6, pp. 271-293, 1999.

[54] K. Govindan and P. Mohaptra, "Trust Computations and Trust Dynamics in Mobile Adhoc Networks: A Survey," *IEEE Communications Surveys and Tutorials*, 2012.

[55] T. Grandison and M. Sloman, "A Survey of Trust in Internet Applications," *IEEE Communication Surveys & Tutorials*, vol. 3, pp. 2-16, 2000.

[56] E. Gray, J. Seigneur, Y. Chen, and C. Jensen, "Trust propagation in small worlds," in *1st International Conference on Trust Management*, 2003, pp. 239-254.

[57] D. Green, "Future of Autonomous Ground Logistics: Convoys in the Department of Defense," United States Army Command and General Staff College, Fort Leavenworth, KS, Monograph 2011.

[58] L. Grossman, "Drone Home," *Rise of the Robots*, pp. 24-33, 2013.

[59] R. Haenni, "Using probabilistic argumentation for key validation in public-key cryptography," *International Journal of Approximate Reasoning*, vol. 38, no. 3, pp. 355-376, 2005.

[60] J. Y. Halpern, D. Samet, and E. Segev, "Defining Knowledge in Terms of Belief: The Modal Logic Perspective," *The Review of Symbolic Logic*, vol. 2, no. 3, pp. 469-487, 2009.

[61] F. M. Ham et al., "Reputation Prediction in Mobile Ad Hoc Networks using RBF Neural Networks," *Engineering Applications of Neural Networks: Communications in Computer and Information Science*, vol. 43, pp. 485-494, 2009.

[62] R. P. Hardie and R. K. Gaye, "Physica," in *The Works of Aristotle vol. 2*, W. D. Ross, Ed. Oxford: Clarendon Press, 1930.

[63] R. Hardin, *Trust and Trustworthiness*. New York, New York: Russell Sage Foundation, 2002.

[64] C. J. Hazard and M. P. Singh, "Intertemporal Discount Factors as a Measure of Trustworthiness in Electronic Commerce," *IEEE Transactions on Knowledge and Data Engineering*, vol. 23, no. 5, pp. 699-712, 2011.

[65] J. L. Herman, *Trauma and Recovery*. New York: Basic Books, 1991.

[66] P. Hieronymi, "The Reasons of Trust," *Australasian Journal of Philosophy*, vol. 86, no. 2, pp. 213-236, 2008.

[67] D. Hume, *Enquiries Concerning the Human Understanding*, 2nd ed., E. Steinberg, Ed. Indianapolis, IN: Hackett Publishing Company, 2011.

[68] IEEE Computer Society. (2002) Foundation for Intelligent Physical Agents (FIPA). [Online]. http://www.fipa.org/specs/fipa00001/

[69] T. Jiang and J. Baras, "Trust Evaluation in Anarchy: A Case Study on Autonomous Networks," in *25th Conference on Computer Communications*, Barcelona, Spain, 2006.

[70] K. Jones, "Second-Hand Moral Knowledge," *Journal of Philosophy*, vol. 96, no. 2, pp. 55-78, 1999.

[71] A. Jøsang, "A Logic for Uncertain Probabilities," *Journal of Uncertainty, Fuzziness, and Knowledge-Based Systems*, vol. 9, pp. 279-311, 2001.

[72] A. Jøsang, "An Algebra for Assessing Trust in Certification Chains," in *Network and Distributed System Security*, San Diego, 1999.

[73] A. Jøsang, R. Ismail, and C. Boyd, "A Survey of Trust and Reputation Systems for Online Service Provision," *Decision Support Systems*, vol. 43, no. 2, pp. 618-644, 2007.

[74] A. Jøsang, S. Marsh, and S. Pope, "Exploring Different Types of Trust Propagation," *Lecture Notes in Computer Science*, pp. 179-192, 2006.

[75] H. Katsuno and A. Mendelzon, "On the difference between updating a knowledge base and revising it," in *Proc. International Conference on Knowledge Representation and Reasoning*, 1991, pp. 387-394.

[76] M. A. Koenig and P. L. Harris, "The Basis of Epistemic Trust: Reliable Testimony or Reliable Sources?," *Episteme*, vol. 4, no. 3, pp. 264-284, 2007.

[77] S. Kripke, "A Completeness Theorem in Modal Logic," *Journal of Symbolic Logic*, vol. 24, no. 1, pp. 1-14, 1959.

[78] R. Levien and A. Aiken, "Attack-Resistant Trust Metrics for Public Key Certification," in *Proc. 7th USENIX Security Symposium*, San Antonio, 1998, pp. 229-242.

[79] J. Li, R. Li, and J. Kato, "Future Trust Management Framework for Mobile Ad Hoc Networks: Security in Mobile Ad Hoc Networks," *IEEE Communications Magazine*, vol. 46, no. 4, pp. 108-114, April 2008.

[80] Z. Li, C. H. Sim, and M. Y. H. Low, "A Survey of Emergent Behavior and Its Impacts in Agent-Based Systems," in *IEEE International Conference on Industrial Informatics*, Singapore, 2006, pp. 1295-1300.

[81] Z. Liu, A. W. Joy, and R. A. Thompson, "A Dynamic Trust Model for Mobile Ad Hoc Networks," in *Proc. 10th IEEE International Workshop on Future Trends of Distributed Computing Systems*, Sushou, 2004, pp. 80-85.

[82] I. S. Livingston and M. O'Hanlon, "Afghanistan Index," Brookings Institute, 2013.

[83] L. Lovasz, "Submodular function and convexity," *Mathematical Programming: The State of the Art*, pp. 235-257, 1983.

[84] N. Luhmann, "Familiarity, Confidence, Trust: Problems and Alternatives," *Trust: Making and Breaking Cooperative Relations*, pp. 94-108, 1988.

[85] D. Majumdar. (2011, December) Iran's Captured RQ-170: How Bad Is The Damage? [Online]. http://www.airforcetimes.com/article/20111209/NEWS/112090311/Iran-s-captured-RQ-170-How-bad-damage-

[86] N. G. Mankiw, *Principles of Microeconomics*, 6th ed. Mason, OH: South-Western Cengage Learning, 2011.

[87] N. Marchang and R. Datta, "Collaborative Techniques for Intrusion Detection in Mobile Ad-Hoc Networks," *Ad Hoc Networks*, vol. 6, pp. 508-523, 2008.

[88] M. Marge et al., "Comparing Heads-Up, Hands-Free Operation of Ground Robots to Teleoperation," in *Robotics: Science and Systems VII*, Los Angeles, 2011.

[89] P. Maynard-Reid and Y. Shoham, "Belief Fusion: Aggregating Pedigreed Belief States," *Journal of Logic, Language, and Information*, vol. 10, no. 2, pp. 183-209, 2001.

[90] D. J. McAllister, "Affect- and Cognition-based Trust as Foundations for Interpersonal Cooperation in Organizations," *Acadmey of Management Journal*, no. 38, pp. 24-59, 1995.

[91] V. McGeer, "Trust, Hope, and Empowerment," *Australasian Journal of Philosophy*, vol. 86, no. 2, pp. 237-254, 2008.

[92] C. McLeod, *Self-Trust and Reproductive Autonomy*. Cambridge, MA: MIT Press, 2002.

[93] C. McLeod. (2011, February) Trust. [Online]. http://plato.stanford.edu/entries/trust/

[94] C. Meyer, *Matrix Analysis and Applied Linear Algebra*. Philadelphia, PA: SIAM, 2000, pp. 661-704.

[95] O. Mistry, A. Gürsel, and S. Sen, "Comparing trust mechanisms for monitoring aggregator nodes in sensor networks," in *Proc. 8th International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, Budapest, 2009, pp. 985-992.

[96] M. Momani, "Trust Models in Wireless Sensor Networks: A Survey," *Recent Trends in Network Security and Applications: Communications in Computer and Information Science*, vol. 89, no. 1, pp. 37-46, 2010.

[97] L. Mui, M. Mohtashemi, and A. Halberstadt, "A Computational Model of Trust and Reputation," in *Proc. 35th Hawaii International Conference on System Sciences*, 2002, pp. 188-197.

[98] R. Mukherjee, B. Banerjee, and S. Sen, "Learning Mutual Trust," *Trust in Cyber-Societies*, pp. 145-158, 2001.

[99] K. W. Nafi, T. S. Kar, Md. A. Hossain, and M. M. A. Hashem, "An Advanced Certain Trust Model Using Fuzzy Logic and Probabilistic Logic Theory," *International Journal of Advanced Computer Science and Applications*, vol. 3, no. 12, pp. 164-173, 2012.

[100] A. Nasipuri, "Mobile Ad Hoc Networks," in *Wireless Networking*. Oxford, UK: Elsevier, 2008, ch. 12, pp. 423-454.

[101] "National Defense Authorization," U.S. Congress, Public Law 106-398, 2001.

[102] NATO Research and Technology Organization, "Multi-Robot Systems in Military Domains," NATO, RTO Technical Report RTO-TR-IST-032, 2008.

[103] E. C. H. Ngai and M. R. Lyu, "Trust and Clustering-Based Authentication Services in Mobile Ad Hoc Networks," in *Proc. 24th International Conference on Distributed Computing Systems Workshops*, 2004, pp. 582-587.

[104] R. Olfati-Saber, J. Fax, and R. M. Murray, "Consensus and Cooperation in Networked Multi-Agent Systems," *Proceedings of the IEEE*, vol. 95, no. 1, pp. 215-233, January 2007.

[105] O. O'Neill, *Autonomy and Trust in Bioethics*. Cambridge: Cambridge University Press, 2002.

[106] D. Pais, "Emergent Collective Behavior in Multi-Agent Systems: An Evolutionary Perspective," Princeton University, Princeton, PhD Thesis 2012.

[107] P. Pappas, "Belief Revision," in *Handbook of Knowledge Representation*, F. van Hrmelen, V. Lifschitz, and B. Porter, Eds. St. Louis, MO: Elsevier, 2007.

[108] D. Parkes and J. Shneidman, "Distributed Implementations of Vickrey-Clarke-Groves Mechanisms," in *3rd international joint Conference on Autonomous Agent and Multi-Agent Systems*, 2004, pp. 261-268.

[109] S. Poslad, M. Calisti, and P. Charlton, "Specifying standard security mechanisms in multi-agent system," in *Workshop on Deception, Fraud, and Trust in Agent Societies, AAMAS 2002*, Bologna, Italy, 2002, pp. 122-127.

[110] N. Potter, *How Can I Be Trusted? A Virtue Theory of Trustworthiness*. Lanham, Maryland: Rowman & Littlefield, 2002.

[111] M. Probst and S. Kasera, "Statistical trust establishment in wireless sensor networks," in *13th International Conference on parallel and Distributed Systems*, 2007, pp. 1-8.

[112] S. D. Ramchurn, D. Huynh, and N. R. Jennings, "Trust in Multi-Agent Systems," *The Knowledge Engineering Review*, vol. 19, no. 1, pp. 1-25, 2004.

[113] W. Ren, R. W. Beard, and D. B. Kingston, "Multi-agent Kalman Consensus with Relative Uncertainty," in *American Control Conference*, Portland, OR, 2005, pp. 1865-1870.

[114] Y. Ren and A. Boukerche, "Modeling and Managing the Trust for Wireless and Mobile Ad Hoc Networks," in *IEEE International Conference on Communications*, 2008, pp. 2129-2133.

[115] S. Ries, "CertainTrust: A Trust Model For Users and Agents," in *Proc. ACM Symposium on Applied Computing*, Seoul, 2007.

[116] Robotic Systems Joint Project Office. (2011, July) Unmanned Ground Systems Roadmap. [Online]. http://www.rsjpo.army.mil/images/UGS_Roadmap_Jul11_r1.pdf

[117] Robotics Collaborative Technology Alliance, "Robotics CTA FY 2012 Annual Program Plan," Army Research Laboratory, 2012.

[118] J. Sabater and C. Sierra, "REGRET: a reputation model for gregarious societies," in *1st International Joint Conference on Autonomous Agents and Multi-Agent Systems*, 2002, pp. 475-482.

[119] E. Schoenherr, "Moving Future Convoy Operations with Convoy Active Safety Technologies (CAST)," *TARDEC Acclerate Magazine*, vol. 4, pp. 74-78, 2009.

[120] S. Sen, "Reciprocity: a foundational principle for promoting cooperative behavior among self-interested agents," in *2nd International Conference on Multi-Agent Systems*, Menlo Park, CA, 1996, pp. 322-329.

[121] J. Sen, P. Chowdhury, and I. Sengupta, "A Distributed Trust Mechanism for Mobile Ad Hoc Networks," in *International Symposium on Ad Hoc and Ubiquitous Computing*, Surathkal, 2008, pp. 62-67.

[122] N. Shachtman. (2011, October) Drone Controls Hit By Cyber Attacks. [Online]. http://www.standupamericaus.org/breaking-news/drone-controls-hit-by-cyber-attacks/

[123] N. Shachtman. (2009, December) Insurgents Intercept Drone Video in King-Size Security Breach. [Online]. http://www.wired.com/dangerroom/2009/12/insurgents-intercept-drone-video-in-king-sized-security-breach/

[124] Y. Shoham and K. Leyton-Brown, "Beyond Belief: Probability, Dynamics, and Intention," in *Multiagent Systems: Algorithmic, Game-Theoretic, and Logical Foundations*. Cambridge: Cambridge University Press, 2009, ch. 14, pp. 421-446.

[125] Y. Shoham and K. Leyton-Brown, "Logics of Knowledge and Belief," in *Multiagent Systems: Algorithmic, Game-Theoretic, and Logical Foundations*. Cambridge: Cambridge University Press, 2009, ch. 13, pp. 393-419.

[126] Y. Shoham and K. Leyton-Brown, "Teams of Selfish Agents: An Introduction to Coalitional Game Theory," in *Multiagent Systems: Algorithmic, Game-Theoretic, and Logical Foundations*. Cambridge: Cambridge University Press, 2009, ch. 12, pp. 367-391.

[127] P. W. Singer, "Advanced Warfare: How We Might Fight With Robots," in *Wired For War*. New York: Penguin Press, 2009, pp. 205-236.

[128] P. W. Singer. (2010, February) The Unmanned Mission. [Online]. http://www.brookings.edu/research/articles/2010/02/22-robot-revolution-singer

[129] S. Singh and S. Thayer, "ARMS (Autonomous Robots for Military Systems): A Survey of Collaborative Robotics Core Technologies and Their Military Applications," Carnegie Mellon University, Tech Report CMU-RI-TR-01-16, 2001.

[130] F. Skopik, D. Schall, and S. Dustdar, "Start Trusting Strangers? Bootstrapping and Prediction of Trust," in *Proc. 10th International Conference on Web Information Systems Engineering*, 2009.

[131] D. P. Spanos, R. Olfati-Saber, and R. M. Murray, "Distributed Sensor Fusion using Dynamic Consensus," in *16th IFAC World Congress*, Prague, 2005.

[132] S. Sternberg, "The Perron-Frobenius Theorem," in *Dynamical Systems*. Mineola, NY: Dover Publications, 2010, ch. 9, pp. 175-195.

[133] J. Sung, L. Guo, R. E. Grinter, and H. I. Christensen, "My Roomba Is Rambo: Intimate Home Appliances," in *LNCS 4717*, 2007, pp. 145-162.

[134] Y. L. Sun and Y. Yang, "Trust Establishment in Distributed Networks: Analysis and Modeling," in *IEEE International Conference on Communications*, Glasgow, 2007, pp. 1266-1273.

[135] L. Teacy, J. Patel, N. Jennings, and M. Luck, "Coping with Inaccurate Reputation Sources: Experimental Analysis of a Probabilistic Trust Model," in *4th International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*, 2005, pp. 997-1004.

[136] L. Teacy, J. Patel, N. Jennings, and M. Luck, "TRAVOS: Trust and reputation in the context of inaccurate information sources," *Autonomous Agents and Multi-Agent Systems*, vol. 12, no. 2, pp. 183-198, 2006.

[137] G. Theodorakopoulos and J Baras, "In Trust Models and Trust Evaluation Metrics for Ad Hoc Networks," *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 2, pp. 318-328, 2006.

[138] R. Tuomela and M. Tuomela, "Cooperation and Trust in Group Context," *Mind and Society*, vol. 4, pp. 49-84, 2005.

[139] U.S Army Research Laboratory. (2012) Network Science Collaborative Technology Alliance. [Online]. http://www.ns-cta.org/

[140] U.S. Department of Defense, "Unmanned Systems Integrated Roadmap, FY2011-2036," Washington D.C., 11-S-3613, 2011.

[141] U.S. Department of the Army, "The United States Army Operating Concept, 2016-2028," Washington D.C., Pamphlet 525-3-1, 2010.

[142] M. Virendra, M. Jadliwala, M. Chandrasekaran, and S. Upadhyaya, "Quantifying trust in mobile ad-hoc networks," in *International Conference on Integration of Knowledge Intensive Multi-Agent Systems*, 2005, pp. 65-70.

[143] Y. Wang, C. Hang, and M. Singh, "A Probabilistic Approach for Maintaining Trust Based on Evidence," *Journal of Artifical Intelligence Research*, vol. 40, pp. 221-267, January 2011.

[144] X. Wang, L. Liu, and J. Su, "RLM: A General Model for Trust Representation and Aggregration," *IEEE Transactions on Services Computing*, vol. 5, no. 1, pp. 131-143, 2012.

[145] Y. Wang and M. Singh, "Formal Trust Model for Multiagent Systems," in *20th International Joint Conference on Artificial Intelligence (IJCAI)*, 2007, pp. 1551-1556.

[146] Y. Wang and J. Vassileva, "Trust and reputation model in peer-to-peer networks," in *Proc. 3rd International Conference on Peer-to-Peer Computing*, Linkoping, 2003, pp. 150-157.

[147] M. Witkowski, A. Artikis, and J. Pitt, "Experiments in building experiential trust in a society of objective trust based agents," *Trust in Cyber-Societies*, pp. 111-132, 2001.

[148] M. Wooldridge, *Reasoning About Rational Agents*. Cambridge, MA: The MIT Press, 2000.

[149] B. Wu, J. Chen, J. Wu, and M. Cardei, "A Survey of Attacks and Countermeasures in Mobile Ad Hoc Networks," *Wireless Network Security Signals and Communication Technology, Part II*, pp. 103-135, 2007.

[150] B. Yu and M. Singh, "An evidential model of reputation management," in *1st International Joint Conference on Autonomous Agents and Multi-Agent Systems*, 2002, pp. 295-300.

[151] B. Yu and M. Singh, "Distributed reputation management for electronic commerce," *Computational Intelligence*, vol. 18, no. 4, pp. 535-549, 2002.

[152] G. Zacharia and P. Maes, "Trust through reputation mechansims," *Applied Artificial Intelligence*, vol. 14, pp. 881-907, 2000.

[153] W. Zhang, S. Das, and Y. Liu, "A Trust Based Framework for Secure Data Agregation in Wireless Sensor Networks," in *3rd IEEE Communication Society of Sensor and Ad Hoc Communications and Networks*, Reston, VA, 2006, pp. 60-69.

[154] K. Ziabari. (2012, December) Another American Drone Captured by Iran: Washington Feels Trepid. [Online]. http://www.globalresearch.ca/another-american-drone-captured-by-iran-washington-feels-trepid/5314601

[155] C. Zouridaki, B. Mark, M. Hejmo, and R. Thomas, "Robust cooperative trust establishment for MANETs," in *4th ACM Workshop on Security of Ad Hoc and Sensor Networks*, 2006, pp. 23-34.